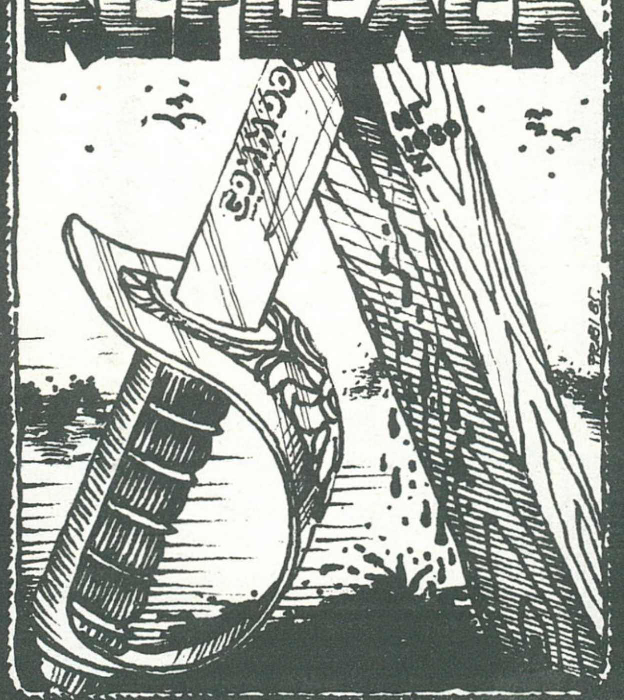
BIT-LET

ötlet SZÁM 1985 AUGUSZTUS 23

REFLEXEK



Reflexeink vannak. Reflexeink, amelyek meghatározzák a dolgokra való első reakcióinkat. Reflexeik vannak az újságíróknak is, s ezekre a reflexekre hallgava hallgatunk több tucat kollégámmal együtt például arról, amiről már hónapok óta szerte az országban beszélnek és tudnak, hogy úgy néz ki, Commodore 16-os gépek lesznek, ha nem is szeptembertől, de ősztől az általános iskolákban. Hallgatunk, mert bár senkinek sem tettünk hűségesküt, de azt mondták nekünk amúgy barátilag, hogy ez még nem publikus, meg még nincs meg a felsőbb döntés. S bár az ilyen baráti kéréseket minden valamire való sajtóban figyelmen kívül hagyják éppen felsőbb szempontra – tudniillik a tájékoztatás szent kötelességére hivatkozva –, de mert reflexeink vannak, hát hallgatunk. Pedig talán a közvéleményt is érdemes lenne nemcsak tájékoztatni, de akár meg is kérdezni, hogy mi a véleménye a lehetőségek ismeretében. Márpedig úgy tűnik, a lehetőség ezúttal egy igazán magyar gép, a Primo és ez az ördöngös BASIC-kel rendelkező Commodore.

A nyári, kicsikét hosszabb szerkesztési és nyomdai átfutási idő következménye, hogy lehetséges, mire olvasóink kezébe kerülnek e sorok, már tele voltak a lapok ezzel a hírrel, mert végre döntés született, aláírták a szerződést, vagy akár már a gépek is úton vannak Budapestre. Most, amikor e sorok íródnak még úgy tudjuk, nincs döntés, nincs szerződés, de erősödik a lehetőség, hogy a párharcból a C–16 hívei kerülnek ki győztesen.

No és mit mondanak minderről a reflexek? Mit mondattak a dologban így-úgy érintettek az elmúlt hónapokban?

Reflexekkel kezdtem, azzal illenék hát folytatni. De, hogy ebben az ügyben mennyire megcsaltak bennünket a reflexek, arra csak egy példa. Kollégám, aki annak idején mélységesen egyetértett azzal a nézettel, hogy a HT helyett ugyanabból a dollárösszegből, amibe a gép alkatrészei kerültek, sokkal érdemesebb lett volna Spectrumokat vagy C–64-eseket osztogatni a középiskoláknak, nos ugyanez a kollégám most mélységesen nem ért egyet a C–16 hívekkel. A hazai ipar védelmét emlegeti – ez is egyfajta reflex –, meg az alkatrészellátást, meg a későbbi újabb gépkontingenst, hogy tudniillik ha most van is, lesz-e legközelebb is elegendő dollárunk, hogy újabb gépeket vegyünk. No és persze a legfőbb reflex is előkerül; minek veszünk valamit kemény valutáért, ha megkapható forintért is. Kollégám érvelésével voltaképpen röviden összegezhettem is azokat a nézeteket, amelyek a Primo mellett tették le voksukat. No és mit mond az ellentábor?

Egyrészt – s ez a legfontosabb – a két gép közti színvonalkülönbségre hivatkozik, amely tagadhatatlanul a színes, nagyfelbontású grafikát BASIC-ben is tudó, sőt saját BASIC rajzoló utasításkészlettel is rendelkező C–16 mellett szól. Emlegetik a két cég között meglévő rutinkülönbséget is, amely mikroelektronikáról lévén szó nem elhanyagolható tényező – s meg is szokott nyilvánulni a gyártott termék megbízhatóságában. Azután arról sem szoktak elfelejtkezni a C–16 hívei, hogy a HT-nél már egyszer sikerült meggyőzni a meggyőzendőket, hogy ez forintért van, azután kiderült, hogy annyi dollár volt abban a forintban, mint égen a csillag. Hmm, hát ebben is van valami.

A két tábor reflexszerű véleményeiben mindkét oldalon sok igazság van. Azt gondoljuk azonban – vagy ha úgy tetszik, szerkesztőségünk azon az állásponton van –, hogy a reflexek helyett ezúttal egyszer végre a józan észnek kellene diadalmaskodni. Márpedig a józan ész azt diktálja, hogy ha egy szemernyi lehetőség is van arra, hogy C–16-os gépek kerüljenek az általános iskolákba, akkor ezt a szemernyi lehetőséget meg kell ragadni.

Úgy reméljük, hogy mire olvassák ezt az írást, a kérdés már nem kérdés. Úgy látjuk, hogy egyre több olyan döntés születik a számítástechnikában is, az oktatásban is, meg a számítástechnika oktatásban is, amely nem reflexszerűen dől el. S úgy reméljük, hogy ha netán e lap megjelenése időpontjában mégsincs döntés, akkor az a tény, hogy az újságíró félretette a reflexeit és megírta azt amit kértek, hogy ne írjon meg, nem árt a döntésnek. Mert ha igen, az is csak egy rossz reflexünket igazolná!

Angyalosi László

BELÜLRŐL

ötlet

HÍROLDAL

Automaták

Az NDK fővárosának újdonsága a közelmúltban üzembe helyezett két takarékpénztári készpénzautomata. A számítógéppel összekapcsolt, mágneskártyával működő automaták éjjel- nappal kiszolgálják az ügyfeleket. A Sparkasse tervei szerint hamarosan újabb berendezéseket helyeznek üzembe Berlinben és az NDK több nagyvárosában.

Computaxa

Mikroszámítógépes taxamétereket szereltek be húsz drezdai taxiba. A korszerű berendezés jelzi a fuvardíjat, különválasztva a várakozási időt és a csomagdíjat, valamint feljegyzi a műszakonkénti fuvarok alapadatait. Az új készüléket az erfurti rádiótechnikai gyár és a drezdai taxivállalat közösen fejlesztette ki.

VT gépek

A Videoton mikroszámítógép-családjának legkisebb tagja a tv-computer. Ennek gyártását ez év második felében kezdik. A már professzionális célokra alkalmas VT–16-ot tavaly kezdték gyártani és az idén 800 készül belőle. A család legnagyobb tagja, a VT–32 a moszkvai magyar jubileumi kiállítás egyik érdekessége és újdonsága volt.

Előrejelzés

Felmérések szerint a múlt évben már mintegy 26 milliárd dollárnak megfelelő értékben értékesítettek személyi számítógépeket. További előrehaladásra lehet számítani. Előrejelzések alapján megállapítható, hogy évi 25 százalékos piaci növekedés várható.

Krimi adatok

A számítógép-korszak létrehozta saját bűnözőtípusát, az okos, intellektuális számítógépbűnözőt. Az újfajta bűnözés ijesztő gyorsasággal terjed. Egy amerikai felmérés szerint a megkérdezett nagy cégek közel ötven százalékánál tapasztaltak tavaly valamilyen mértékű számítógépes bűntettet, ami általában bizonyos pénzösszegek saját célra történő átutaltatásából állt. Az ismertté vált bűnügyek okozta kár körülbelül fél milliárd dollárra rúg. A fel nem derített eseteket is beleértve az összeget mintegy hárommilliárd dollárra becsülik.

Mikro áramkör

Számottevő különbségek vannak az elektronizáció terén a közel azonos fejlettségű nagy nyugat-európai országok között is. Az NSZK, Nagy-Britannia és Franciaország ipari üzemeiben végzett felmérés adatai szerint az NSZK-ban alkalmazzák a legszélesebb körben az elektronika eredményeit: az üzemek 47 százaléka hasznosítja a mikroáramköröket a gyártáshoz és 13 százalékuk be is építi ezeket a végtermékbe. Ugyanezek az adatok Angliában 43 és 10 százalék, míg Franciaországban csak 35 és 6 százalék.

Ultrahanggal

Többek között a távfűtéses házak, illetve lakások hőenergia-fogyasztásának mérésére, díjkiszámítására fejlesztettek ki a nyugatnémet Siemens vállalatnál egy ultrahangos műszert. Ez ultrahanggal méri a lakásba befolyó víz mennyiségét és a be- és kijövő víz hőmérsékletkülönbségét. Az így szerzett adatokból egy, a műszerrel egybeépített mikroszámítógép számolja ki a fogyasztói díjakat.

Osztrák trend

A számítógépesítésben nyugat-európai partnereitől viszonylagosan elmaradt Ausztriában tavaly mintegy 70 ezer házi számítógépet és 12 ezer professzionális mikroszámítógépet értékesítettek. Ebben az évben már, az előrejelzések szerint 110 ezer házi és 20 ezer professzionális mikrogép talál majd gazdára.

Szabásminta

Két és félmillió forintos megtakarítást terveznek a Szegedi Ruhagyárban a számítógépes szabásoptimalizálás bevezetésével. A számítógép segítségével a férfiöltönyök kiszabásánál a kb. százféle öltönyrészt úgy helyezik el a szöveten, hogy a legkisebb hulladék képződjön.

Célgép

Az Inovion nevű, Utah állambeli cég egy különleges képfeldolgozó célgép forgalmazását jelentette be. A gép ára 3500 dollár, alkalmas arra, hogy kameráról vagy más eszközről származó képeket digitalizáljon. A szoftver tartalmaz olyan lehetőségeket, amelyek segítségével a digitalizált képet módosítani, „festeni" lehet. A Personal Graphics System nevű gép képernyője 512x480 képpont, az elérhető színárnyalatok száma a képben 250 000.

Döntés

Fontos döntést hozott a közelmúltban a Szovjetunió Kommunista Pártjának Politikai Bizottsága. Ez év szeptember elsejétől az ország valamennyi középiskolájában meg kell kezdeni a számítástechnikai alapismeretek oktatását.

ÚJ!

A Jarogate nevű brit mikroszámítógép gyártó cég új többfelhasználós személyi számítógép forgalmazásába kezdett. A számítógép neve: Sprite.
Központi egysége egy 80286-os mikroprocesszor, alapkiépítésében 512 kbyte RAM-ot, egy 21 Mbyte-os Winchester lemezegységet és egy 790 Kbyte kapacitású floppy lemezmeghajtót tartalmaz. Ez a konfiguráció két terminál csatlakoztatását teszi lehetővé, ára 4995 angol font.
Az operatív tár bővíthető 2 Mbyte-ra, a merevlemezes háttértár 150 Mbyte-ra. Maximális kiépítésben 16 terminál csatlakoztatható hozzá. A legnagyobb kiépítés ára kb. 15 ezer angol font. Az operációs rendszer konkurrens CP/M 3.1, amely rendelkezik PC–DOS emulációval. A benchmark kísérletek azt bizonyítják, hogy a Sprite gyorsabb a PC AT-nál, vagy a Motorola 68000-es alapú Pinnacle-nél is. A kép tanúsága szerint a gyártmány – külső megjelenését tekintve – emlékeztet a VT 16-osra.



DATA Generáltól

Új 32 bites miniszámítógéppel jelent meg a piacon a Data General cég. Az Eclipse MV–1000 SX névre hallgató gép mintegy harminc százalékkal magasabb teljesítményű a cég előző, hasonló típusától. Alapkiépítésben, négy darab tárolóval és rendszerszoftverrel ára 223 ezer dollár. Az új eszközhöz 32 megabyte-os tárolóbővítéseket és szoftverbővítéseket ajánlanak.

Horoszkóp

Mérföldkőhöz érkeztünk. Néhány évvel ezelőtt rácsodálkoztunk, ha főleg szórakoztató hetilapokban horoszkóppal találkozhattunk. Sokan komolytalansággal is vádolták az ilyen lapokat. Aztán arról olvastunk, hogy nyugaton horoszkóp készítésére számítógépet vesznek igénybe. Ezen már csak mosolyogtunk. S íme most itt a rideg valóság: Szentendrén néhány forintért mikroszámítógéppel jósoltathat magának a hétvégi kiránduló. S, még mondja valaki, hogy nem terjed nálunk a „modern technika és annak „modern" alkalmazása.

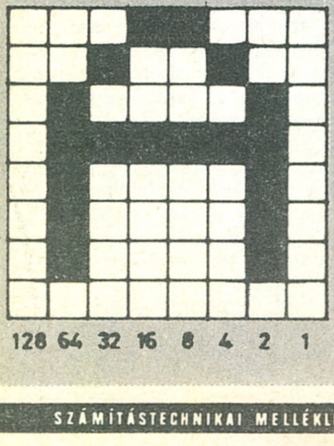
Szempanasz!

A számítógépekhez használatos képernyős terminálok egyre gyakrabban okoznak különféle szempanaszokat. Az esetek számának növekedése természetesen a technika általánossá válásával magyarázható. A betegek legtöbbször gyengén látásra, égető fájdalomra, könnyezésre, gyors szemkifáradásra panaszkodnak, melynek oka elsősorban az asztigmatizmus okozta helytelen fénytörés és a helyiségek nem kielégítő megvilágítása. Megállapították, hogy ezeken kívül panaszt okoz az, hogy a terminál kezelőinek sokat kell a képernyőre és az adatokat, szöveget tartalmazó papírlapokra nézniük, és a változó távolság miatt a pupilla hosszú időn át folyamatosan kitágul és összehúzódik.
Jelentős befolyásoló tényező a képernyők színe, fényereje is.



VC20

A VC–20 vallatóját olvasva kissé én is elcsodálkoztam, talán csak a dokumentáció csapnivalóságában egyeztem meg az olvasottakkal. Amiről most én szeretnék írni az egy sprite-(manó) tervezés VC–20-on, de mindjárt azt is hozzáteszem, hogy ez közel sem hasonló a C–64-eséhez. Mint tudjuk, a két gép a memóriaméreten és a képernyőfelbontáson túl abban különbözik, hogy a VC–20 nem tud sprite-okat tervezni. Ha olyat nem is tud, mint a C–64, de azért figurákat ezzel a géppel is lehet tervezni, mint a programból ki fog derülni.
Tulajdonképpen a dolognak a lényege a jelgenerátorban van. A gép jelgenerátora ROM-ban van elhelyezve a 32768-as memóriahelyen kezdődik és 4 K hosszú. Ez a következőképpen oszlik meg:
1 K – nagybetűk
1 K – inverz nagybetűk
1 K – kisbetűk
1 K – inverz kisbetűk
A karakterek elhelyezése a már az Ötletből is ismert módon történik: egy karakter 8 byte-on helyezkedik el, a byte-ok binárisan vannak kódolva aszerint, hogy a megadott hely be van-e sötétítve, vagy sem. Nézzük meg példaként az A betű elhelyezését:



1. byte : 24
2. byte : 36
3. byte : 66
4. byte : 126
5. byte : 66
6. byte : 66
7. byte : 66
8. byte : ø

A byte-ok sorfolytonosan vannak elhelyezve a memóriában, a karakterek a képernyőkódoknak megfelelő sorrendben következnek egymás után. Tehát: @, A, B, C,... (gépkönyvben megtalálhatók). Az interpreter egy billentyű lenyomásakor a következő módon találja meg a karaktert a jelgenerátorban: kiolvassa a videomátrixból az adott karakterhez tartozó jelindexet, majd ezt 3-szor balra shifteli (szorzás 8-cal). Az eredményt hozzáadja a 36869 címen tárolt jelcella tartomány start címéhez. A kapott címen találja meg az interpreter a karaktert jellemző 8 byte-ot és ezt viszi ki képernyőre. Ha át akarjuk helyezni a karakterkészletet, akkor a 36869 címen található startcímet kell módosítani. Eredetileg ezen a címen 240 található, ez 32 K jelgenerátor címnek felel meg (nagybetűk). 24 K–34 K-nak felel meg (kisbetűk). Áthelyezéshez a következő értékeket kell megadni:
252–4 K (4096)
253–5 K (5120)
254–6 K (6144)
255–7 K (7168)
Előzőleg azonban a karaktereket meg kell tervezni (a 8 byte-ot) és a megfelelő memóriaterületre be kell vinni, majd beírni egy POKE 36869, megfelelő érték. Ezután a megfelelő billentyűk elvesztik eredeti funkciójukat, és az általunk megtervezett karaktert viszik ki a képernyőre. A sprite, hogy jól látható legyen kétszeres nagyságúra tervezhető. Az eredeti karakterek 8x8-as jelmátrixban helyezkednek el, de van egy lehetőség 16x8-as jelmátrixok kiválasztására is. Ezt a 36867-es memóriacímen levő byte-tal lehet. Ennek bitjei:
0: 0= 8x8-as jelmátrix
1=16x8-as jelmátrix
1–6: kiteszik a videómátrix sorszámát
7: raszterérték egy eleme.

Ezek után nézzük a programot:
A működése a következő:
A program a 3,5 K-s VC–20 utolsó 0,5 K-ját szabaddá teszi a tervezendő ábrák számára.
Ha 16x8-as ábrákat tervezünk ez a terület akkor is 32 db számára elegendő.
Indítás után kirajzolja az ábrát, ahova meg kell terveznünk a sprite-ot. Ezután nyomjuk be a CTRL+RVS ON gombokat, így inverz üzemmódba kerülünk. Majd a kurzorral felmegyünk a raszterhálóra és a space gomb nyomogatásával berajzoljuk a megfelelő ábrát. Ha mindezekkel készen vagyunk rávisszük a kurzort a RUN 80-as sorra és RETURN-t nyomunk. A gép ekkor helyezi el az ábrát a memóriába, majd vagy újrakezdjük elölről az egészet vagy kilépünk a programból és megnézzük, hogy sikerültek ábráink. Ha készen vagyunk, a programot ki is törölhetjük a memóriából, és dolgozhatunk tovább az általunk tervezett karakterekkel (sprite-okkal). A sprite-okat a játékprogramoknál megismert módszerekkel mozgathatjuk továbbra is, rájuk a megfelelő billentyű kódjával hivatkozhatunk. A gép kikapcsolása előtt ajánlatos a megfelelő memóriarészt szalagra menteni, hogy ne kelljen újból tervezni.

A program részletesebben:
0 sor: felszabadít 0,5 K-t, és kitörli ezt a területet (7168–7680)
1–5 sor: képernyőt töröl, színtárolót állít be
10–40: kilép az első programból
80: kezdődik a 2. program.
100: A két program közötti paraméter-átadásra az 5400-as memóriarekesz szolgál. Ebben tároljuk, hogy hányadik karakternél tartunk, és ezt adjuk hozzá az új karaktertároló kezdőcíméhez (7168). Egyszerű változóval ez nem oldható meg, mivel a RUN 80 minden változót töröl.
110–160: az előbbi ábrába rajzolt pontok kiolvasása, byte-okká alakítása, elhelyezése a memóriában.
170–210: Van-e még karakter?
Ha van: 5400-as cím növelése 1-gyel, visszatérés.
Ha nincs: átváltás az új karaktertárolóra (POKE 36869,255)
átváltás 16x8-as jelmátrixra (POKE 36867,153)
12 sorunk lesz a képernyőn! ←
Aki csak 8x8-as jelmátrixokat akarja alkalmazni, a következő módosításokat kell eszközölnie:
10 FOR I = 1 TO 8
100 . . . . . : Q = PEEK (5400) * 8+7168
110 FOR I = 1 TO 8
200 törölni: POKE 36867,153
Mindenkinek sok sikert és jó kísérletezést kívánok a program kipróbálásához.

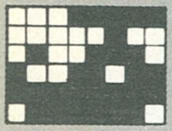
Békési Károly
villamos üzemmérnök, Kulturális
Központ Marcali, Széchenyi u. 3. 8700

```
0 POKE56,28:POKE5400,0:FORI=7168TO7680:POKEI,0:NEXT
1 PRINT"♫":K=7687:REM"♫"=SHIFT+CLR
5 FORI=38400TO38906:POKEI,0:NEXT
10 FORI=1TO16
15 POKEK-1,103
20 FORJ=KTOK+7:POKEJ,122:NEXT
30 K=K+22
40 NEXT
50 FORI=1TO18:PRINT:NEXT:REM"⌐"=CRSR↑
60 PRINT"RUN80⌐⌐⌐"                 "
70 END
80 PRINT"⌐⌐
90 PRINT"DOLGOZOM!!!"
100 K=7687:Q=PEEK(5400)*16+7168
110 FORI=1TO16
120 FORJ=0TO7
130 IFPEEK(K+7-J)=160THENC=C+2↑J
140 NEXT
150 POKEQ,C:Q=Q+1:K=K+22:C=0
160 NEXT
170 PRINT"⌐VAN MEG?(I/N)"
180 GETA$:IFA$=""THEN180
190 IFA$="I"THENPOKE5400,PEEK(5400)+1:GOTO1
200 IFA$="N"THENPOKE36869,255:POKE36867,153:PRINT"♫":END
210 GOTO180
```



HT 1080 Z

A program lehetővé teszi 8 db 8x6 raszterpontból (4x2 karakter) alakzat tárolását, ezek közül egyidejűleg 4 db megjelenítését a képernyőn, a C–64 gép sprite-jaihoz hasonlóan. Egy lehetséges alakzat pl:



AØ, 82, BB, 93	16Ø, 13Ø, 187, 147
AD, BE, BD, 9F	173, 19Ø, 189, 159

Az alakzat (sprite) tárolása a karakterek ASCII kódja alapján történik. Kiszámítási elv:

1	2
4	8
16	32

+128 (Amelyik be van töltve, azok értékét összeadjuk és még 128-at hozzáadunk:

A fenti sprite kódjait mellé is írtam (hexadecimálisan és decimálisan is). Ezeket a kódokat kell betölteni a 8 blokk valamelyikébe. A sprite-ok helyét a képernyőn 2–2 byte-on kell megadni a következőképpen: kiszámítjuk, hogy a bal felső karakter melyik memóriacellába kell hogy kerüljön: K=(sorok száma) ∗64+oszlopok száma+15360 (itt a sorok és oszlopok karakterméretre vonatkoznak). Majd az így kapott értéket felbontjuk 2 byte-ra: az alsó byte KAND 255 művelettel kapható, a felső byte: INT(K/256) művelettel. Az így kapott két számot kell betölteni a megfelelő két memóriacellába. A sprite-ok helyének ilyen módon való meghatározására valók a 0–7. rendszerváltozók a programban. A 8. rendszerváltozó megfelelő bitjének 1-re állításával megjeleníthető bármelyik sprite, 0-ra állításával pedig kikapcsolható. A 9. és 10. rendszerváltozó megfelelő 4–4 bitjén meghatározható, hogy egy-egy sprite alakja melyik blokkban tárolt képnek legyen megfelelő. Így tehát, ha négy helikoptert akarnak megjeleníteni és mozgatni a képernyőn, nem szükséges négy különböző blokkba betölteni a fent említett kódokat, hanem elég egybe, a blokkmutatót viszont mind a négy sprite-nál arra állítjuk. Ha egy sprite alakját ki akarjuk cserélni egy másikkal, akkor sem kell a kódokat mozgatni, csak a blokkmutatót átállítani.

A rendszerváltozók a 7A70H=31344 címtől kezdődnek: x=31344.

x. memóriacella: 0. sprite helyének alsó byte-ja
x+1. memóriacella: 0. sprite helyének felső byte-ja
x+2. memóriacella: 1. sprite helyének alsó byte-ja
x+3. memóriacella: 1. sprite helyének felső byte-ja
x+4. memóriacella: 2. sprite helyének alsó byte-ja
x+5. memóriacella: 2. sprite helyének felső byte-ja
x+6. memóriacella: 3. sprite helyének alsó byte-ja
x+7. memóriacella: 3. sprite helyének felső byte-ja
x+8. memóriacella:
0. bit = 0. sprite engedélyezése – letiltása
1. bit = 1. sprite engedélyezése – letiltása
2. bit = 2. sprite engedélyezése – letiltása
3. bit = 3. sprite engedélyezése – letiltása
4–7. bit nem használt
x+9. memóriacella:
alsó 4 bit: 0. sprite blokkmutatója
felső 4 bit: 1. sprite blokkmutatója
x+10. memóriacella:
alsó 4 bit: 2. sprite blokkmutatója
felső 4 bit: 3. sprite blokkmutatója

Az értékek Basicből POKE-utasítással állíthatók. Az alakzatok imént említett módon kiszámított kódjait a következő memóriaterületekre kell POKE-olni:

0. blokk: 7A00H=31232–7A07H=31239
1. blokk: 7A08H=31240–7A0FH=31247
2. blokk: 7A10H=31248–7A17H=31255

3. blokk: 7A18H=31256–7A1FH=31263
4. blokk: 7A20H=31264–7A27H=31271
5. blokk: 7A28H=31272–7A2FH=31279
6. blokk: 7A30H=31280–7A37H=31287
7. blokk: 7A38M=31288–7A3FH=31295

A sor elején levő szám a blokkok sorszáma ezek közül a megjeleníteni kívántat kell a 9–10. rendszerváltozóba beírni. A program használatát segít megérteni az a Basic bemutatóprogram, amit közöltem. Tehát a Basic program nem tartozik hozzá a sprite-programhoz, csupán egy demonstrációs program. A sprite-ok használata kissé bonyolultnak tűnhet, de előnyei miatt azt hiszem, érdemes használni a Basic játékprogramokhoz. Hiszen nem kell foglalkoznunk azzal, hogy az alakzat milyen képernyőterület előtt halad el, ugyanis átsiklik felette anélkül, hogy kárt tenne benne. A Demo programból láthatóan még egymáson is nyugodtan át tudnak „úszni" a sprite-ok. És sokkal gyorsabb is, mintha Basic-ben állandóan letörölnénk és újabb helyre kirajzolnánk egy ilyen alakzatot. Itt nem kell a törléssel sem foglalkozni, azt is elvégzi a rutin. A rutin meghívása a már említett műveletsorral történik. A program használakor vigyázzunk, hogy a sprite-ok helye ne kerüljön a képernyőn kívülre! (Ha el akarjuk tüntetni, tiltsuk le.) A program 7A00–7B80-ig használja a memóriát. Ezért 1000 byte-nál többet ne foglaljunk le stringek számára! (Vagy bekapcsoláskor a READY-re 32230-at válaszoljunk.) Ezzel sok sikert kívánok a program használatához.

Zsoldos Zsolt Tata, Eötvös Gimnázium III. osztályos tanulója

```
SPRITE PROGRAM HT-1080Z GEPRE

BETOLTES: 7A80H=31360 CIMTOL

MEGHIVAS: POKE16526,128:POKE16527,122:U=USR(1)   UTASITASSORRAL

2A 40 7A EB 21 42 7A CD AB 7A 2A 4A 7A EB 21 4C 7A CD AB 7A
2A 54 7A EB 21 56 7A CD AB 7A 2A 5E 7A EB 21 60 7A CD AB 7A
C3 BC 7A 01 04 00 ED B0 01 3C 00 EB 09 EB 01 04 00 ED B0 C9
2A 70 7A 22 40 7A 11 42 7A CD EF 7A 2A 72 7A 22 4A 7A 11 4C
7A CD EF 7A 2A 74 7A 22 54 7A 11 56 7A CD EF 7A 2A 76 7A 22
5E 7A 11 60 7A CD EF 7A C3 FE 7A 01 04 00 ED B0 01 3C 00 09
01 04 00 ED B0 C9 3A 72 7A CB 47 28 09 2A 70 7A 3A 79 7A CD
51 7B 3A 78 7A CB 4F 28 11 2A 72 7A 3A 79 7A CB 1F CB 1F CB
1F CB 1F CD 51 7B 3A 78 7A CB 57 28 09 2A 74 7A 3A 7A 7A CD
51 7B 3A 78 7A CB 5F 28 11 2A 76 7A 3A 7A 7A CB 1F CB 1F CB
1F CB 1F CD 51 7B C3 76 7B 06 0F A0 87 87 87 5F 16 7A CD 65
7B 01 3C 00 09 CD 65 7B C9 06 04 1A 4E CB 79 20 02 0E 80 B1
77 23 13 10 F2 C9 C9 78 7A 3E 0F A6 21 42 7A CD B9 7B CB 7F

KESZITO: ZSOLDOS ZSOLT   TATA,EOTVOS J. GIMN.

DEMONSTRACIOS. PROGRAM  A  SPRITE-PROGRAMHOZ:

100 CLEAR1000:CLS:FORY=1TO15:A$=STRING$(20-Y,32)+CHR$(164)+STRING$(Y*3,191)+CHR$
(180)+"  ":PRINTA$:NEXTY
110 FORI=31232TO31239:READA:POKEI,A:NEXT
120 FORI=31240TO31263:READA:POKEI,A:NEXTI
130 R=31344:POKER+8,15:POKER+9,19:POKER+10,2:POKER+1,61:POKER+3,60:POKER+5,60:PO
KER+7,60
140 FORZ=0TO2:FORX=0TO60:POKER,136-X:POKER+2,X+20:POKER+4,2*X+10:POKER+6,240-X*4
150 POKE16526,128:POKE16527,122:U=USR(1)
160 NEXTX:POKER+1,Z+60:POKER+3,Z+60:POKER+5,62-Z:POKER+7,61-Z:NEXTZ:GOTO140
170 DATA151,131,131,171,181,176,176,186
180 DATA176,181,186,176,131,151,171,131
190 DATA128,130,187,147,173,190,189,159
200 DATA 166,166,166,166,153,153,153,153
```



GÉPIKÓD II.

			N	V	B	D	I	Z	C
0	BRK	STOP			1				
1,a	ORA (zx). a	A=A OR PEEK(PEEK(z+x)+256 * PEEK(z+x+1))	+					+	
5,a	ORA z,a	A=A OR PEEK (z)	+					+	
6,a	ASL z,a	POKE z. (PEEK(z) * 2)AND 255	+					+	+
8	PHP	S(SP) P SP SP-1							
9,a	ORA a	A A or a	+					+	
10	ASL	A (A * 2) AND 255	+					+	+
13,a,b	ORA c,a+256 * b	A A OR PEEK (c)	+					+	
14,a,b	ASL c,a+256 * b	POKE c. (PEEK(c) * 2)AND 255	+					+	+
16,a	BPL a	IF NBIT 0 THEN GOTO(PC+a+256 * (a -127))							
17,a	ORA (z)y,a	A=A OR PEEK(PEEK(z)+256 * PEEK (z+1)+y)	+					+	
21,a	ORA z+x,a	A=A OR PEEK (z+x)	+					+	
22,a	ASL z+x,a	POKE z+x. (PEEK(z+x) * 2)AND 255	+					+	+
24	CLC	CBIT 0							0
25,a,b	ORA c+y,a+256 * b	A=A OR PEEK(c+y)	+					+	
29,a,b	ORA c+x,a+256 * b	A=A OR PEEK (c+x)	+					+	
30,a,b	ASL c+x,a+256 * b	POKE c+x (PEEK(c+x) * 2)AND 255	+					+	+
32,a,b	JSR c,a+256 * b	GOSUB c							
33,a	AND (zx),a	A=A AND PEEK(PEEK(z+x)+256 * PEEK (z+x+1))	+					+	
36,a	BIT z,a	ZBIT=(A AND PEEK(z)=0) NBIT=-(PEEK(z)AND 128):							
		VBIT=-(PEEK(z)AND 64)	+	+				+	
37,a	AND z,a	A=A AND PEEK(z)	+					+	
38,a	ROL z,a	POKEz. ((PEEK(z) * 2)AND 255)+CBIT	+					+	+
40	PLP	SP=SP+1 : P=S(SP)	+	+	+	+	+	+	+
41,a	AND a	A=A AND a	+					+	
42	ROL	A=((A * 2)AND255)+CBIT	+					+	+
44,a,b	BIT c,a+256 * b	ZBIT=(A AND PEEK(c)=0) NBIT= (PEEK(c)AND							
		128) : VBIT=-(PEEK(c)AND 64)	+	+				+	
45,a,b	AND c,a+256 * b	A=A AND PEEK (c)	+					+	
46,a,b	ROL c,a+256 * b	POKE c,((PEEK(c) * 2)AND 255)+CBIT	+					+	+
48,a	BMI a	IF NBIT=1 THEN GOTO(PC+a+256 * (a 127))							
49,a	AND (z)y,a	A=A AND PEEK(PEEK (z)+256 * PEEK(z+1)+y)	+					+	
53,a	AND z+x,a	A=A AND PEEK (z+x)	+					+	
54,a	ROL z+x,a	POKE z+x, ((PEEK(z+x) * 2)AND 255)+CBIT	+					+	+
56	SEC	CBIT=1							1
57,a,b	AND c+y,a+256 * b	A=A AND PEEK (c+y)	+					+	
61,a,b	AND c+x,a+256 * b	A=A AND PEEK (c+x)	+					+	
62,a,b	ROL c+x,a+256 * b	POKE c+x,((PEEK(c+x) * 2)AND 255)+CBIT	+					+	+
64	RTI	CONT	+	+	+	+	+	+	+
65,a	EOR (zx),a	A=A EOR PEEK(PEEK(z+x)+256 * PEEK(z+x+1))	+					+	
69,a	EOR z,a	A=A EOR PEEK (z)	+					+	
70,a	LSR z,a	POKE z. INT (PEEK (z)/2)	0					+	+
72	PHA	S (SP)=A:SP=SP-1							
73,a	EOR a	A=A EOR a	+					+	
74	LSR	A=INT (A/2)	0					+	+
76,a,b	JMP c,a+256 * b	GOTO c							
77,a,b	EOR c,a+256 * b	A=A EOR PEEK (c)	+					+	
78,a,b	LSR c,a+256 * b	POKE c,INT (PEEK(c)/2)	0					+	+
80,a	BVC a	IF VBIT=0 THEN GOTO (PC+a+256 * (a > 127))							
81,a	EOR (z)y,a	A=A EOR PEEK(PEEK(z)+256 * PEEK (z+1)+y)	+					+	
85,a	EOR z+x,a	A=A EOR PEEK (z+x)	+					+	
86,a	LSR z+x,a	POKE z+x,INT (PEEK(z+x)/2)	0					+	+
88	CLI	IBIT=0					0		
89,a,b	EOR c+y,a+256 * b	A=A EOR PEEK (c+y)	+					+	
93,a,	EOR c+x,a+256 * b	A=A EOR PEEK (c+x)	+					+	
94,a,b	LSR c+x,a+256 * b	POKE c+x,INT (PEEK(c+x)/2)	0					+	+
96	RST	RETURN							
97,a	ADC (zx),a	A=A+PEEK(PEEK(z+x)+256 * PEEK(z+x+1))+CBIT	+	+				+	+
101,a	ADC z,a	A=A+PEEK(z)+CBIT	+	+				+	+
102,a	ROR z,a	POKE z, INT(PEEK(z)/2)-128 * (CBIT=1)	+					+	+
104	PLA	SP=SP+1 : A=S(SP)	+					+	
105,a	ADC a	A=A+a+CBIT	+	+				+	+
106	ROR	A=INT (A/2)-128 * (CBIT=1)	+					+	+
108,a,b	JMP (c).a+256 * b	GOTO (PEEK(c)+256 * PEEK(c+1))							
109,a,b	ADC c,a+256 * b	A=A+PEEK(c)+CBIT	+	+				+	+
110,a,b	ROR c,a+256 * b	POKE c,INT (PEEK(c)/2-128 * (CBIT-1)	+					+	+
112,a	BVS a	IF VBIT=1 THEN GOTO (PC+a+256 * (a -127))							

113,a	ADC (z)y,a
117,a	ADC z+x,a
118,a	ROR z+x,a
120	SEI
121,a,b	ADC c+y,a+2
125,a,b	ADC c+x,a+2
126,a,b	ROR c+x,a+2
129,a	STA (zx),a
132,a	STY z,a
133,a	STA z,a
134,a	STX z,a
136	DEY
138	TXA
140,a,b	STY c,a+256
141,a,b	STA c,a+256
142,a,b	STX c,a+256
144,a	BCC a
145,a	STA (z)y,a
148,a	STY z+x,[illegible]
149,a	STA z+x,a
150,a	STX z+y,a
152	TYA
153,a,b	STA c+y,[illegible]+2
154	TXS
157,a,b	STA c+x,a+2
160,a	LDY a
161,a	LDA (zx),a
162,a	LDX a
164,a	LDY z,a
165,a	LDA z,a
166,a	LDX z,a
168	TAY
169,a	LDA a
170	TAX
172,a,b	LDY c,a+256
173,a,b	LDA c,a+256
174,a,b	LDX c,a+256
176,a	BCS a
177,a	LDA (z)y,a
180,a	LDY z+x,a
181,a	LDA z+x,a
182,a	LDX z+y,a
184	CLV
185,a,b	LDA c+y,a+2
186	TSX
188,a,b	LDY c+x,a+2
189,a,b	LDA c+x,a+2
190,a,b	LDX c+y,a+2
192,a	CPY a
193,a	CMP(zx), a
196,a	CPY z,a
197,a	CMP z,a
198,a	DEC z,a
200	INY
201,a	CMP a
202	DEX
204,a,b	CPY c,a+256 *
205,a,b	CMP c,a
206,a,b	DEC c,a+256 *
208,a	BNE a
209,a	CMP (z,[illegible]a
213,a	CMP z+x,[illegible]
214,a	DEC z+x,a
216	CLD
217,a,b	CMP c+y[illegible]+2
221,a,b	CMP c+x,a+2
222,a,b	DEC c+x,a+2
224,a	CPX a
225,a	SBC (zx),a
228,a	CPX z,a
229,a	SBC z,a
230,a	INC z,a
232	INX
233,a	SBC a
234	NOP
236,a,b	CPX c,a+256 *
237,a,b	SBC c,a+256 *
238,a,b	INC c,a+256 *
240,a	BEQ a
241,a	SBC (z)y,a
245,a	SBC z+x,a
246,a	INC z+x,a
248	SED
249,a,b	SBC c+y,a+2
253,a,b	SBC c+x,a+2
254,a,b	INC c+x,a+2

		N	V	B	D	I	Z	C
y,a	A=A+PEEK(PEEK(z)+256*PEEK (z+1)+y)+CBIT	+	+				+	+
x,a	A=A+PEEK(z+x)+CBIT	+	+				+	+
x,a	POKE z+x,INT(PEEK(z+x)/2-128*(CBIT=1)	+					+	+
	IBIT=1					1		
y,a+256*b	A=A +PEEK(c+y)+CBIT	+	+				+	+
x,a+256*b	A=A +PEEK(c+x)+CBIT	+	+				+	+
x,a+256*b	POKE c+x,INT(PEEK(c+x/2)-128*(CBIT=1)	+					+	+
),a	POKE(PEEK(z+x)+256*PEEK(z+x+1),A							
	POKE z,y							
	PKOE z,A							
	POKE z,x							
	Y=(Y-1)AND 255	+					+	
	A=x	+					+	
+256*b	POKE c,y							
+256*b	POKE c,A							
+256*b	POKE c,x							
	IF CBIT=0THEN GOTO(PC+a+256x(a > 127))							
y,a	POKE (PEEK(z)+256*PEEK(z+1)+y),A							
x,	POKE z+x,y							
x,a	POKE z+x,A							
y,a	POKE z+y,x							
	A=y	+					+	
y,a+256*b	POKE c+y,a							
	SP=X	+					+	
x,a+256*b	POKE c+x,A							
	y=a	+					+	
),a	A=PEEK(PEEK(z+x)+256*PEEK(z+x+1))	+					+	
	x=a	+					+	
	y=PEEK(z)	+					+	
	A=PEEK(z)	+					+	
	x=PEEK(z)	+					+	
	y=A	+					+	
	A=a	+					+	
	x=A	+					+	
+256*b	y=PEEK(c)	+					+	
+256*b	A=PEEK (c)	+					+	
+256*b	x=PEEK (c)	+					+	
	IF CBIT=1 THEN GOTO(PC+a+256*(a > 127))							
y,a	A=PEEK(PEEK(z)+256*PEEK(z+1)+y)	+					+	
x,a	y=PEEK (z+x)	+					+	
x,a	A=PEEK (z+x)	+					+	
y,a	x=PEEK (z+y)	+					+	
	VBIT=0		0					
y,a+256*b	A=PEEK(c+Y)	+					+	
	x=SP	+					+	
x,a+256*b	y=PEEK(c+X)	+					+	
x,a+256*b	A=PEEK (c+x)	+					+	
y,a+256*b	x=PEEK (c+y)	+					+	
	CBIT=-(y >=a):ZBIT=-(y=a):NBIT=(y-a)AND 128	+					+	+
, a	N=PEEK(PEEK(z+x)+256*PEEK(z+x+1)):CBIT=-(A >=N): ZBIT=-(A=N):NBIT=(A-N)AND 128	+					+	+
	N=PEEK(z):CBIT=-(y>=N):ZBIT=-(y=N):NBIT=(y-N)AND 128	+					+	+
	N=PEEK(z):CBIT=-(A >=N):ZBIT=-(A=N):NBIT=(A-N)AND 128	+					+	+
	POKE z, (PEEK(z)-1)AND 255	+					+	
	y=(y+1) AND 255	+					+	
	CBIT=-(A >=a):ZBIT=-(A=a):NBIT=(A-a)AND 128	+					+	+
	x=(x-1)AND 255	+					+	
+256*b	N=PEEK(c):Bitállítás a 196-os utasitással egyező	+					+	+
	N=PEEK(c):Bitállitás a 197-es utasitással egyező	+					+	+
+256*b	POKE c,(PEEK(c)-1)AND 255	+					+	
	IF ZBIT=0 THENGOTO(PC+a+256*(a > 127)							
y,a	N=PEEK(PEEK(z)+256*PEEK(z+1)+y): Bitállitás a 197-es utasitással egyező	+					+	+
x,a	N=PEEK(z+x):Bitállitás a 197-es utasitással egyező	+					+	+
x,a	POKE z+x,(PEEK(z+x)-1)AND 255	+					+	
	DBIT=0				0			
y,a+256*b	N=PEEK(c+y): Bitállitás a 197-es utasitással egyező	+					+	+
x,a+256*b	N=PEEK(c+x): Bitállitás a 197-es utasitással egyező	+					+	+
x,a+256*b	POKE c+x,(PEEK(c+x)-1)AND 255	+					+	
	CBIT=-(x >=a):ZBIT=-(x=a):NBIT=(x-a) AND 128	+					+	+
),a	A=((A-PEEK(PEEK(z+x)+256*PEEK(z+x+1)))AND 255)+(CBIT=0)	+	+				+	+
	N=PEEK(z):CBIT=-(x>=N):ZBIT=-(x=N):NBIT=(x-N)AND 128	+					+	+
	A=((A-PEEK(z))AND255)+(CBIT=0)	+	+				+	+
	POKE z,(PEEK(z)+1)AND 255	+					+	
	x=(x+1) AND 255	+					+	
	A=((A-a)AND 255)+(CBIT=0)	+	+				+	+
	Üres utasitás							
+256*b	N=PEEK(c):Bitállitás a 228-as utasitással egyező	+					+	+
+256*b	A=((A-PEEK(c))AND 255)+(CBIT=0)	+	+				+	+
+256*b	POKE c, (PEEK(c)+1)AND 255	+					+	
	IF ZBIT=1 THEN GOTO (PC+a+256*(a > 127))							
y,a	A=(((A-PEEK(PEEK(z)+256*PEEK(z+1)+y))AND 255)+(CBIT=0)	+	+				+	+
x,a	A=((A-PEEK(z+x))AND 255)+(CBIT=0)	+	+				+	+
x,a	POKE z+x,(PEEK(z+x)+1)AND 255	+					+	
	DBIT=1				1			
y,a+256*b	A=((A-PEEK(c+y))AND 255)+(CBIT=0)	+	+				+	+
x,a+256*b	A=((A-PEEK(c+x))AND 255)+(CBIT=0)	+	+				+	+
x,a+256*b	POKE c+x, (PEEK(c+x)+1) AND 255	+					+	

BIT-LET

A Commodore VC–20 gépi kódú programozásához a Motorola 6502-es processzor utasításkészletét kell használnunk. A táblázat első oszlopában találjuk az utasításnak megfelelő kód számát, továbbá, hogy az utasítás használja-e az utána következő egy vagy két byte-ot. „a" az első byte tartalmát, „b" a második byte tartalmát jelenti. A táblázat 2. oszlopában találjuk az utasítás 3-betűs mnemonikját, amely az utasítás hatására vonatkozó angol nyelvű rövidítés Pl. LDA az a "load in accumulator" rövidítése, lefordítva „tölts az akkumulátorba" utasítást jelenti. Ebben az oszlopban találjuk azt a címzési formát is, amely az utasítás után következő egy vagy két byte tartalmának felhasználására vonatkozik,

Pl. a z+x, a címzési forma jelentése: az utasítás utáni byte tartalmából egy címet állít elő úgy, hogy hozzáadja a X-regiszter tartalmát. Az utasítás az „a" értékét zérólaposnak tekinti, ezért nincs szüksége egy második byte-ra. A c+x, a256*+b címzési forma ezzel szemben jelzi, hogy a cím nem zérólapos, ezért szükség van mindkét bytera is, amelynek tartalmát az a+256*b képlettel használja fel a processzor, végül a teljes címet még az X-regiszter tartalmának hozzáadásával állítja elő. A harmadik oszlopban találjuk az utasítás BASIC-nyelvű megfeleltetését vagy leírását. Amelyikben nincs POKE utasítás, azokat egy-egy rövid programban kísérletképpen kipróbálhatjuk. Pl.: 29,0,16=ORA c+x, 0+256*16 (Az akkumulátort a logikai OR kapcsolatba állítja a c+x címzési formával előállított címmel.)

```
10 A+25:X+40:a+0:b+16
20 A=A OR PEEK (a+256*b+x)
30 PRINT A
```

A negyedik oszlopban találjuk az utasításoknak a feltételregiszter jelző bitjeire gyakorolt hatását.

Jelölések:
- A = akkumulátor
- x = x-regiszter
- y = y-regiszter
- PC = utasítás számláló
- a = utasítás után következő byte tartalma
- b = utasítás után következő második byte tartalma
- S = veremtár
- SP = veremtármutató
- P = feltételregiszter
- (kisbetűs) z = zérólapos cím = a
- (kisbetűs) c = két byte-os cím = a + + 256*b
- (nagybetűs) N, V, B, D, I, Z, C = feltételregiszter bitjei
- + = bitet állíthat mindkét féle módon
- 0 = nullára állítja csak
- 1 = egyre állítja csak
- N = az utasításhoz felhasznált közbenső adat

Inkvizítorok kerestetnek!

A C–16-os és Sinclair QL gépek vallatására készülünk. Kérjük, hogy aki már kellő tapasztalatokkal rendelkezik valamelyik gép használatát, programozását illetően – s szívesen részt venne e vallatáson – írjon nekünk. Röviden írja le, hogy mióta dolgozik a géppel, hogy előtte milyen gépeken szerzett tapasztalatokat, s hogy a C–16, illetve a QL programozásban meddig jutott. Kérjük, hogy a jelentkezők nevük és pontos címük mellett, ha lehetséges telefonszámot is közöljenek, ahol megtalálhatók.

– A Szerk. –

ötlet

SPECTRUM PASCAL

A Hisoft cég egy jól megírt, kiforrott Pascal fordító programot forgalmaz a ZX Spectrumhoz HP4T néven. A program törzsét a Z80 mikroprocesszorra fejlesztették ki, és adaptálták minden Z80-al működő mikroszámítógépre, s egyéb Z80-as rendszerekre is. Ebből következően ennek a programnak hatalmas nagy a piaca, ezért fordíthattak a kidolgozására szokatlanul nagy gondot. Ezt használata során rögtön tapasztalhatjuk: kellemes vele dolgozni, és rendkívül hatékony fordító. Ennek egyik oka a hatékonyságában rejlik: az előállított tárgykód sebessége akár több nagyságrenddel is meghaladhatja az ugyanarra a feladatra írt BASIC programét. Természetesen a konkrét sebességarány erősen függ a feladat jellegétől. Másrészt a HP4T a file kezeléstől eltekintve majdnem teljes PASCAL nyelvet valósít meg, amely sokkal alkalmasabb komoly programfejlesztésre, mint a BASIC. Jóval több programozási eszközt tartalmaz, ezzel segíti a felhasználót a különböző feladatok kívánalmaihoz alkalmazkodni. Talán a legfontosabb eltérés, hogy a BASIC-ben nincs lehetőség a magasszintű nyelvekben egyébként megszokott eljárás (szubrutin) hívásra. A BASIC GOSUB utasítása a legprimitívebb gépi kódú szubrutin hívással ekvivalens, pl. a Z80 feltétel nélküli CALL-jával. Meg kell jegyeznünk, hogy már a harmadik generációs nagygépeknek is fejlettebb gépi szubrutin hívásuk van. A PASCAL viszont lehetővé teszi a formális – aktuális paraméter cserét és az eljárásra nézve lokális azonosítók használatát. Ez biztosítja, hogy zökkenőmentesen vegyük át mások eljárásait vagy saját korábbi program részleteinket. Mint látni fogjuk, a HP4T különösen támogatja az eljárás gyűjtemény létrehozását és új programjainkba való rugalmas, könnyen kezelhető beépítését.

Azok számára, akik a ZX Spectrumon szeretnék használni a PASCAL nyelvet, nagy segítséget jelenthet egy közelmúltban megjelent könyv:

Varga Imre–Ury László: Pascal Spectrumra és Commodore 64-re

LSI Alkalmazástechnikai Tanácsadó Szolgálat Budapest, 1985.

A könyv túlnyomórészt a HP4T-vel foglalkozik. A C–64-re három rendszert ismertet vázlatosan a számos létező közül. Ezek mindegyikére érvényes a szerző állítása: egyik sem „igazi" PASCAL. A Spectrum PASCAL-t teljes részletességgel ismerteti, a HP4T által megvalósított nyelv szintaxisát, szemantikáját és a fordító használatát egyaránt. A szerző tömören, de világosan fogalmaz, az egész könyv rendkívül alapos munka benyomását kelti. Bátran merem ajánlani a PASCAL-ban kezdőknek és haladóknak egyaránt. (feltéve, hogy meg tudják fizetni). A néhány értelemzavaró sajtóhibán kívül csak egyetlen érdemi kifogásom van. Az eljárások és függvények formális-aktuális paramétercseréjéről a 38. oldalon a következő áll: „A paraméterhivatkozás, akár az ALGOL-ban, kétféle lehet: érték, illetve név szerinti." Ez az állítás félrevezető, mert a PASCAL-nak ugyan valóban kétféle paraméterhivatkozási lehetősége van, de csak az egyiket tekinthetjük azonosnak a megfelelő ALGOL hivatkozással. Az érték szerinti paraméterátadásnak csak a szintaxisa különbözik a két nyelvben, ezeket működés szempontjából azonosnak tekinthetjük. A PASCAL másik paraméterhivatkozása viszont a FORTRAN (egyetlen) cím szerinti hivatkozásával ekvivalens, ennél lényegesen összetettebb az ALGOL név szerinti hivatkozása. Erre a tényre maguk a PASCAL-nyelv definiálói is felhívják a figyelmet K. Jensen–N. Wirth: PASCAL felhasználói kézikönyv és jelentés című könyvükben. A felhasználói kézikönyv bevezetésében ismertetik a főbb eltéréseket más magasszintű nyelvektől, itt kihangsúlyozzák, hogy a PASCAL-ban név szerinti hívás nincs. A Varga-könyvben ez csak egy elnevezésbeli tévedés, mert a cím szerinti hivatkozás lényegét később helyesen magyarázza.

A HP4T adaptálása a ZX Spectrumra több lépésben ment végbe, egymással felülről kompatibilis változatok jelentek meg. Hazánkban három verzió közismert, ezek a kibocsátás sorrendjében a következők:

verziószám	név	kezdőcím	hossz
V1.4	HP4S	24598	21105
V1.5	HP4T15M	24598	21462
V1.6	HP4TM16	24598	21609

A HP4S nem tud microdrive-ot kezelni, és a magnón is más formátumot használ, mint a későbbi változatok. Jó tudni, hogy a HP4S-sel felvett forrásnyelvű szöveget be tudják olvasni a későbbi változatok, de fordítva nem. Az 1.5-ös és 1.6-os változatok – mint ahogy a nevükben levő M betű is mutatja – képesek mircodrive-ot kezelni. Az összes műveletet, amit a HP4S magnón végezni tud ezek a változatok microdrive-val is végre tudják hajtani. A HP4T15M több hibát tartalmaz a gyártó cég saját bevallása szerint is, ezért használat ellenjavallt.

A Hisoft hirdetéseiben régóta olvashatjuk, hogy dolgoznak egy olyan változaton, amely a PASCAL szabványos file kezelését megvalósítja a microdrive-on. Nincs információm arról, hogy ez a változat kapható-e már a piacon. (Ha valakinek megvan, kérem, feltétlenül értesítsen.)

Mint már említettem, a HP4T majdnem teljes PASCAL-nyelvet valósít meg. A lényeges korlátozások a következők:

(1) File típus nincs. Ez egy igen gyors és megbízható háttértár (pl. Winchester diszk) hiányában nem túl erős korlátozás.

(2) Formális és aktuális paraméter nem lehet függvény vagy eljárás.

(3) Record változórészt nem tartalmazhat. Azaz a record deklarációban nem szerepelhet a CASE többirányú elágazó utasítás.

(4) Mutató típus nem mutathat mutatóra.

(5) A halmaztípus csak a 0..255 intervallumon vagy ennek egy részintervallumán definiálható Például a VAR halmaz 1: SET OF 256..267. deklaráció nem megengedett. (Viszont az alaphalmaz maximális mérete kellemesen nagy.)

Némi ellensúlyozásként a HP4T kiterjeszti a CASE többirányú elágazó utasítás lehetőségeit: a címkével ellátott utasításlista végére az END lezárás helyett felvehetünk egy ELSE ágat. Erre csak akkor kerül a vezérlés, ha a szelektor kifejezés értéke a lista egyik címkéjével sem egyezik. Ha valamelyik listaelem végrehajtódott, akkor az ELSE ág kimarad és a CASE után következő utasításra kerül a vezérlés.

Tekintsük át röviden a HP4T felépitését és használatát.

Három fő részt különböztethetünk meg: szövegszerkesztő, fordító, futásidő rutinok.

A szövegszerkesztő (editor) teljesen felülbírálja a BASIC billentyűkezelést. Ennek következtében le kell mondanunk az automatikus kulcsszavas üzemmódról, pedig ez azoknak, akik megszokták, nagy segítséget ad a gyors és hatékony programfejlesztéshez. Azt hiszem, a két nyelv gyökeresen eltérő kulcsszókészlete miatt nem lehetett jobb megoldást találni, mint hogy mindent karakterenként ki kell írni. A programszöveg javítása kényelmesen végezhető, változatos lehetőségek állnak rendelkezésünkre. (Pl. sorszám szerinti javítás vagy karakterlánc-keresés és szükség esetén csere!) Itt jegyzem meg, hogy ugyanez az editor található meg a Hisoft cég másik kiváló programtermékében a GENS3 assembler fordítóban. Egyetlen kényelmetlen tulajdonsága van ennek a szerkesztőnek, hogy egy sor javítása közben (edit módban) a DELETE gomb nem töröl, csak a kurzort lépteti eggyel vissza. Törlésre két lehetőségünk van: vagy az E (edit) parancs K (kill) alparancsával töröljük azt a karaktert amelyen a kurzor áll, vagy az I alparanccsal insert módba lépünk, ahol a DELETE gomb eredeti funkciójában működik. A K lenyomásakor nem látjuk, hogy mit töröltünk, ez több karakter esetén zavaró. A program beírása közben élő DELETE funkciót edit módban túl körülményesen érhetjük csak el: előtte insert almódba kell lépnünk, onnan törlés után ENTER-rel kell visszatérnünk edit módba. Azt hiszem, a HP4T editora még ezzel a kis hibával is hatékonyabban használható szövegjavításra mint az eredeti Basic editor.

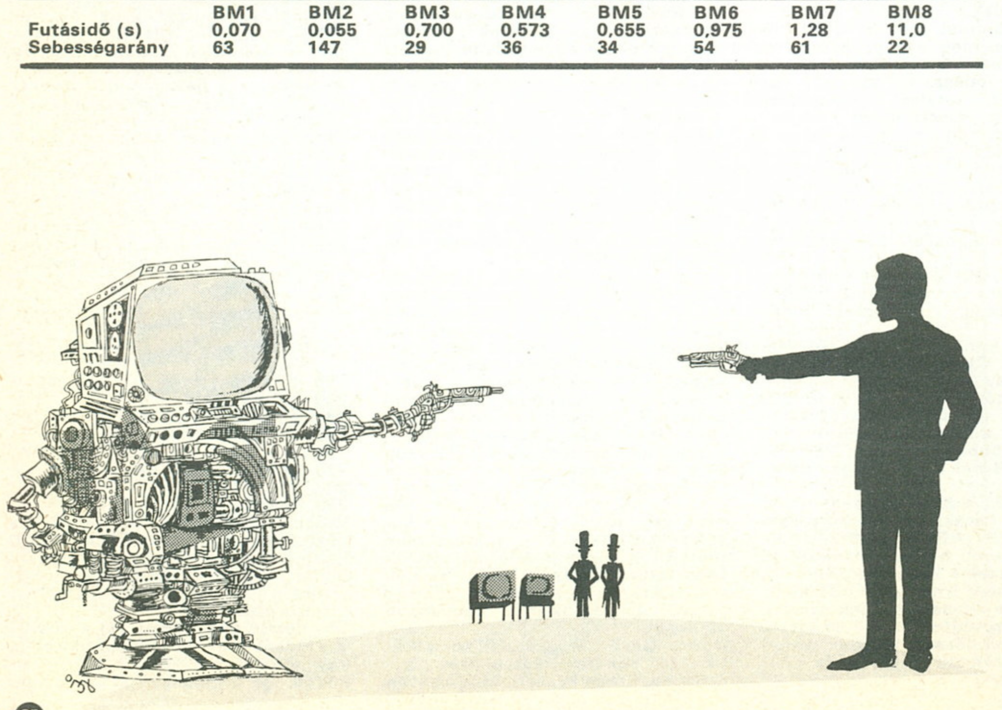
Editor parancsokkal tárolhatjuk a forrásnyelvű szöveget magnón vagy microdrive-on. A kivitelre a PUT parancs szolgál. Az így kivitt programot a GET paranccsal olvashatjuk be. A magnó és microdrive közti választás névválasztással közvetve történik. Ha a név első karaktere 1 és 8 közötti számjegy, a második pedig kettőspont (:) akkor a választott periféria az első karakterrel megadott sorszámú microdrive; a tényleges név pedig az utolsó hat karakter lesz. Ez a konvenció minden háttértár-kezelési műveletre érvényes. Magnóról lehetőség van üres névvel olvasni, ekkor az első megtalált file töltődik be, bizonyos megszorításokkal. Hasznos a B parancs, amely visszaadja a vezérlést a Basic interpreternek. BASIC-ből három címen indíthatjuk a HP4T-t: hidegstart (a forrásnyelvű szöveg törlődik): 24598; melegstart (a forrásnyelvű program megmarad): 24603; ha beolvasáskor nem hagytuk automatikusan elindulni, akkor az első indítás címe 24608. Ha az első indítás már megtörtént, akkor a 24608 címre adott vezérlés teljes memóriatörléshez vezet.

A HP4T fordító részét kétféle üzemmódban használhatjuk. Programfejlesztés közben kényelmes a PASCAL nyelvű szöveget és ennek lefordított tárgykódját egyszerre a memóriában tartani (a fordító mellett). Ha a program mérete ezt lehetővé teszi, akkor használhatjuk a compile parancsot, melynek alakja: Cn. Hatására a forrásnyelv program az n-edik sortól kezdve lefordítódik. Ha n-et elhagyjuk, akkor az első létező sortól indul a fordítás. A szintaktikus hibákról egyértelmű üzeneteket kapunk fordítás közben. A hibás sor után megáll a fordító, megjelöli a hiba valószínű helyét és egy számmal kódolt jelzést ad a hiba jellegéről. Ekkor az E lenyomásával javíthatjuk az adott sort vagy a P lenyomásával az előzőt. Minden más billentyűre folytatódik a fordítás, de ilyenkor a hibajelzések többsége hamis

a korábbi sor értelmezésének elmaradása miatt, tehát nem érdemes tovább engedni a fordítást.
A HP4T kihasználja, hogy a PASCAL-nyelv szintaxisa lehetővé teszi az egymenetes fordítást. A fordítási idő meglepően csekély, ha a fordítás közbeni listázást letiltjuk, akkor 100 soronként kb. 1 másodperc. A letiltást egy fordítási opcióval végezhetjük, amely megjegyzés formájában a programban bárhol elhelyezhető. Alakja: {$ L –}, ahol $ jelzi a fordítónak, hogy opció következik, az L a listázásra utal, a – pedig a letiltásról intézkedik: az opció után levő sorok nem listázódnak. (Az alapértelmezés L+, azaz minden sor listázódik.) Ha az L – kiadása után szintaktikus hibát talál a fordító, akkor a hibás sort kiírja, a javítás ugyanúgy végezhető, mint L+ esetben. A futás közbeni ellenőrzések vezérlésére még számos opciót használhatunk ugyanilyen szintaxissal.
Még egy hasznos fordítási opciót szeretnénk megemlíteni, amelynek hatására a fordítandó szöveg a háttértárról olvasódik be, a memóriában való tárolás nélkül. Alakja: {$ F név}, ahol a név egy magnóra, vagy microdrive-ra elmentett forrásnyelvű program file neve. Magnóra az ilyen file-okat speciális formátumban, a W paranccsal kell elhelyezni, microdrive-on erre nincs szükség, ott erre a célra is a P parancsot kell használni. Ezzel olyan hosszú programokat is futtatni tudunk, amelyek tárgykódjukkal együtt nem férnek el a memóriában. Másrészt ez az opció ad lehetőséget, hogy új programjainkba kényelmesen beépíthessük a gyakran használt programrészleteket.
A másik fordítói üzemmódban olyan tárgykódot állíthatunk elő, amely a HP4T memóriában való jelenléte nélkül futtatható. Erre a translate parancs szolgál. Alakja: Tn,,f (A HP4S-nél csak Tn írható, f automatikusan az előzőleg más háttértár művelettel beállított érték.) Ennek hatására lefordítódik a forrásprogram, ez után az „OK?" üzenet jelenik meg. Ha erre Y-al (yes) válaszolunk, akkor a tárgykód és a futásidő rutinok elmentődnek a háttértáron. Az OK? kérdésre azért van szükség, mert a folyamat a fordító törlődésével jár, viszont ha bármilyen Y-tól különböző billentyűvel válaszolunk, akkor semmisnek minősül a transzlálási kísérlet.
A HP4T által megvalósított – a szabványos PASCAL-ban nem szereplő – utasítások közül, csak kettőre szeretnék kitérni, ezek egyfajta elemi file-kezelést valósítanak meg. Tetszőleges memóriaterület kivihető háttértárra a TOUT (név, cím, hossz) eljárással. A névnek pontosan nyolc karakteres stringnek kell lennie (mert típusa ARRAY(1..8) OF CHAR), ez lesz a file neve. (Az első karakterrel itt is kiválaszthatjuk a microdrive-okat). A másik két paraméterrel azt adjuk meg, hogy melyik címtől hány byte-ot kívánunk kivinni. A kimentett információt visszaolvashatjuk a TIN (név, cím) eljárással. Itt nincs szükség a hossz megadására, mert azt úgyis tartalmazza a file fej, a cím megadásával viszont máshová olvashatjuk be az információt, mint ahonnan kivittük. Ezek az eljárások két speciális függvény segítségével használhatók kényelmesen. Ezek lehetővé teszik, hogy tetszőleges típusú változókat (tehát pl. tömböket is) úgy vigyünk ki és olvassunk vissza, hogy közben nem veszünk tudomást a konkrét kezdőcímükről és méretükről. Ezek az ADDR (V) és a SIZE (V), ahol V egy tetszőleges típusú változó azonosítója. Az ADDR megadja azt az egész számot, amely a változó kezdőcíme a memóriában, s SIZE pedig a hosszát byte-okban. Használatukra példa:
TOUT ('A vektor', ADDR(A), SIZE (A));
TIN ('A vektor', ADDR (B));
Először egy A vektor nevű file-ba kivittük az A azonosítójú változó tartalmát, később visszaolvastuk a B változóba.
Végül szeretnék néhány konkrét adatot közölni a fordító hatékonyságáról. A PASCAL/BASIC feladattól függő sebességarány alsó határa 15 körül van. Ezt csak rövid programokkal lehet elérni, amelyeknél még nem nyilvánul meg a BASIC-nek az a hátránya, hogy a futásidőigény növekszik pusztán a programszöveg terjedelmének gyarapodásától. Meg kell jegyeznem, hogy már ez a sebességarány is minőségi különbséget jelenthet egy program gyakorlati alkalmazásánál. Például egyáltalán nem mindegy, hogy olyan függvénygrafikonok előállítására, melyekből több tucatra van szükségünk, 70 másodpercet kell várnunk vagy csak ötöt. Komolyabb feladatokra írt programok esetében a sebességarány jelentősen megnövekszik. Pl. többváltozós függvények minimumának számításánál a sebességarány eléri az ötvenet. Hasonló értékeket tapasztaltunk a közönséges differenciálegyenlet-rendszert megoldó negyedrendű Runge-Kutta integrátornál is. A nagyon sokszorosan egymásbaskatulyázott ciklusokat felhasználó programok esetén 100 körüli sebességarányt is tapasztaltak. Az általam ismert legszélsőségesebb példát a már említett Jensen-könyvben szereplő primszámgeneráló program adta. A program 1-től 10000-ig állítja elő a primszámokat az eratosztenészi szita algoritmussal. A HP4T-vel megvalósított változat futásideje egy jellemző pontig 4 mp volt. A BASIC változat ugyaneddig a pontig 40 percig futott, tehát a sebességarány 600. Ezt elsősorban azzal magyarázhatjuk, hogy a PASCAL halmazműveletek nagyon gyorsak, és a BASIC-ből teljesen hiányzik ez az adatábrázolási lehetőség.
A relatív adatok után ismertetnék két abszolút sebességadatot is, ahol a megfelelő BASIC-program futásideje nem ismert. Egy általam írt Gyors Fourier Transzformáció helytakarékos változata egy 256 pontból álló mérési mintát 21 másodperc alatt transzformál. Ez a transzformációhoz szükséges szinusz és koszinusz értékeket mindannyiszor kiszámolja, ahányszor felhasználja őket. Az időtakarékos változat ezeket az értékeket a főprogramból egy tömbben átveszi (ezzel ugyanannyi memóriát foglal el, mint a hasznos információval). Ennek a futásideje 11 másodperc. Ez a közelmúlt minigépein tisztán gépi kódban fejlesztett programokéval azonos teljesítmény! Kíváncsiságból elvégeztem a BENCHMARK tesztet a HP4TM16-al. A ciklusok futásszámát 32-, illetve 16-szorosára kellett növelnem, hogy egyáltalán mérni tudjam az időt. A táblázatban közölt adatok 1000 lefutásra vannak átszámolva. A második sorban a PASCAL-BASIC sebességarány szerepel. Ezzel kapcsolatban meg kell jegyeznem, hogy az én gépemen a BASIC futásidők néhány százalékkal kisebbek voltak, mint a BIT-LET 1984. januári számában olvasható idők. Valószínű, hogy a két mérés eltérő verziószámú ZX Spectrumokon történt. A sebességarányt a saját méréseim alapján számoltam. A BASIC-ról PASCAL-ra fordításnak több lehetősége is van, ezért közlöm az általam használt programok listáit is.

Tarnóczi Tibor

	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8
Futásidő (s)	0,070	0,055	0,700	0,573	0,655	0,975	1,28	11,0
Sebességarány	63	147	29	36	34	54	61	22



```
>L
  10 PROGRAM BENCHMARK1;
  20 VAR K:INTEGER;
  30 BEGIN
  40  PAGE;
  50  WRITELN('S');
  60  FOR K:=1 TO 32000 DO;
  70  WRITELN('E');
  80 END.
```

```
>L
  10 PROGRAM BENCHMARK2;
  20 LABEL 1;
  30 VAR K:INTEGER;
  40 BEGIN
  50  PAGE;
  60  WRITELN('S');
  70  K:=0;
  80 1:K:=K+1;
  90  IF K<32000 THEN GOTO 1;
 100  WRITELN('E');
 110 END.
```

```
>L
  10 PROGRAM BENCHMARK3;
  20 LABEL 1;
  30 VAR K,A:INTEGER;
  40 BEGIN
  50  PAGE;
  60  WRITELN('S');
  70  K:=0;
  80 1:K:=K+1;
  90  A:=K DIV K*K+K-K;
 100  IF K<16000 THEN GOTO 1;
 110  WRITELN('E');
 120 END.
```

```
>L
  10 PROGRAM BENCHMARK4;
  20 LABEL 1;
  30 VAR K,A:INTEGER;
  40 BEGIN
  50  PAGE;
  60  WRITELN('S');
  70  K:=0;
  80 1:K:=K+1;
  90  A:=K DIV 2*3+4-5;
 100  IF K<16000 THEN GOTO 1;
 110  WRITELN('E');
 120 END.
```

```
>L
  10 PROGRAM BENCHMARK8;
  20 LABEL 1;
  30 VAR K:INTEGER;
  40  A,B,C:REAL;
  50 BEGIN
  60  PAGE;
  70  WRITELN('S');
  80  K:=0;
  90 1:K:=K+1;
 100  A:=SQR(K);
 110  B:=LN(K);
 120  C:=SIN(K);
 130  IF K<1000 THEN GOTO 1;
 140  WRITELN('E');
 150 END.
```

```
>L
  10 PROGRAM BENCHMARK5;
  20 LABEL 1;
  30 VAR K,A:INTEGER;
  40  PROCEDURE EMPTY;
  50   BEGIN
  60   END;
  70 BEGIN
  80  PAGE;
  90  WRITELN('S');
 100  K:=0;
 110 1:K:=K+1;
 120  A:=K DIV 2*3+4-5;
 130  EMPTY;
 140  IF K<16000 THEN GOTO 1;
 150  WRITELN('E');
 160 END.
```

```
>L
  10 PROGRAM BENCHMARK6;
  20 LABEL 1;
  30 VAR K,A,L:INTEGER;
  40  M:ARRAY[1..5] OF INTEGER;

  50  PROCEDURE EMPTY;
  60   BEGIN
  70   END;
  80 BEGIN
  90  PAGE;
 100  WRITELN('S');
 110  K:=0;
 120 1:K:=K+1;
 130  A:=K DIV 2*3+4-5;
 140  EMPTY;
 150  FOR L:=1 TO 5 DO;
 160  IF K<16000 THEN GOTO 1;
 170  WRITELN('E');
 180 END.
```

```
>L
  10 PROGRAM BENCHMARK7;
  20 LABEL 1;
  30 VAR K,A,L:INTEGER;
  40  M:ARRAY[1..5] OF INTEGER;

  50  PROCEDURE EMPTY;
  60   BEGIN
  70   END;
  80 BEGIN
  90  PAGE;
 100  WRITELN('S');
 110  K:=0;
 120 1:K:=K+1;
 130  A:=K DIV 2*3+4-5;
 140  EMPTY;
 150  FOR L:=1 TO 5 DO
 160   M[L]:=A;
 170  IF K<16000 THEN GOTO 1;
 180  WRITELN('E');
 190 END.
```



Szomorú nap volt V. 28. (kedd). Számos más gondom mellett az is emlékezetessé tette, hogy ismét egy olyan könyvet vettem kézbe, amelynek megírását régóta tervezem. (Sajnos nem ez az első alkalom, például Csákány–Vajda: Játékok számítógéppel és Székely Gábor: Paradoxonok a véletlen matematikájában olvasásakor éreztem már hasonlót.)

Örömmel üdvözlöm ugyanakkor az első nagy példányszámú vásárolható, magyar nyelvű BASIC-tankönyvet. A BASIC-nyelv magyarországi térhódításának 3–4. évében végre megszületett a nagy hiányt pótló mű: Pál Zsuzsanna–Révbíró Tamás: Hetedhét C–16 (és C–64). Kiadó és terjesztő: Novotrade RT. Az alábbiakban arról lesz szó, miért tetszett (és persze a teljesség kedvéért, mennyiben nem tetszett), illetve miért tartom elsőnek a műfajban.

Tetszett – a stílusa, hangvétele. A cím és az embléma azt sugallja, hogy gyerekeknek szánták a szerzők. Sok 9–10 éves tanulóval kísérleteztem, kivétel nélkül folyamatosan tudták olvasni. Ezen felbátorodva nem szakember felnőttek kezébe adtam. Őket jobban zavarta a fogyatékos nyomdatechnika, de szintén értették, sőt ravaszabb keresztkérdésekre is elfogadható válaszokat kaptam. Ugyanakkor félek, a Tankönyvkiadó nem osztaná véleményemet, hasonló stílussal eddig csak a Zsolnay-kísérlet mert próbálkozni. Külön szerencse, hogy „oktett", „helyőr", „modul" és társai nem vétettek át!

Tetszett – pici fenntartásokkal – ezekről később – az arányérzéke. Példásan egyensúlyoz egyszerű, illetve bonyolult és fontos dolgok között, ami a folyamatos olvashatóság alapfeltétele. Zseniális az „angol–magyar helyesírás" című gordiuszi csomó szétvágása a 106. oldalon (az oldalszámozás írásomban: 100∗kötet+oldalszám):

„... – a gépen csak az angol ábécé betűi találhatók meg. Ezt azonban eléggé könnyű megszokni; csak egy dolog fontos: hogy emiatt senki se felejtse el a magyar helyesírást!"

Figyelmes, hogy szerzők gondoltak az angolul nem (tökéletesen) tudó olvasóra is: jelentés, helyesírás, sőt néhol angol–amerikai eltérés (pl.: 147. oldal).

Tetszett – Lényegében a tagolás. Kellemes a terjedelem, és a három kötet. A huszonegy rész sorrendje is nagyjából indokolt, és kb. külön „órákként" (15–30 percekként) vehetők a részek.

Tetszett, hogy például a változók (azonosítók) többször is előjöttek, és száraz prédikáció helyett szinte észrevétlenül szed fel az olvasó némi „matematikát". A grafika is fokozatosan tárul fel, aláhúzva a tankönyv és kézikönyv közötti különbséget. Élvezettel olvastam az „Ezzel az információval egyelőre nem sokat tudunk kezdeni, de..." kezdetű mondatot a 113. oldalon.

–... sőt lenyűgözött a megjelenés sebessége. Csak sejtjük még, hogy C–16 lesz szeptembertől az iskolákban, és íme, máris. Hej, ha a HT... Ez a tény még a nyomdatechnika hibáit is menti.

Tetszett – a GOTO – FOR – IF utasítások bemutatásának ilyen sorrendje. Nem állítom, hogy ez az üdvözítő, de imponált, hogy valaki ezt is megpróbálta. Tapasztalataink szerint ez gyorsabb, simább út, hátránya, hogy az IF lényege kevésbé rögződik – viszont a FOR jobban.

Tetszettek – emberszabású példák, és főleg jópofa, egyszerű összefoglaló kérdések – ahol vannak. Ideértem a szándékos hibákat is (pl.: 122., 218. oldal).

Tetszettek – a „motivált oktatás" című didaktikai bravúrok. A 121. oldalon például soronként törölhetünk egy programot (ez már ismétlés), majd a „fáradt" olvasó örömmel látja a NEW parancsot – és meg is jegyzi egy életre

Nagyon tetszett a könyv! Elsősorban az egész, de tetszett sok apró ötlet, amelyek lényegesek.

Tetszett a szerzőpáros összetétele: A számítógépet éppen ízelgető nem-műszaki (Révbíró T.) saját szavaival írja le azt a folyamatot, amint egy szakember (Pál Zs.) bevezeti őt a programozás világába.

(Szubjektiv) fenntartásaim lényegtelen apróságokkal kapcsolatban vannak:

1. Az INST/DEL billentyű használatát későbbre tenném, hiszen bizonyos esetekben (") meglepő hatása lehet.

2. Kicsit korán szerepel túl sok billentyű. C=gombot elhalasztanám néhány fejezettel.

3. Vitathatónak érzem a 3. fejezetet. Külön választanám a LOAD – és később a SAVE parancsot, sőt sokkal későbbre tenném a VERIFY, "$", OPEN-nel kapcsolatos tudnivalókat.

4. 156–157. oldalakon van olyan utasítás, a listákban, ami még nem szerepelt (IF, READ, GET). Ezt illene említeni, ha az ellenkezőjét hangsúlyozzák a 155. oldalon!

5. Picit többet írnék az IF-ről, véleményem és tapasztalataink szerint ez a legproblémásabb utasítás. Szerzők mindössze kétszer annyi fejezetet szánnak erre (10. és 17.), mint az órára.

6. Már szidtam az olvashatóságot. Konkrétabban: Nincs ékezetes szerkesztő? Nincs szebb printer? Már újságban (pl. BIT-LET) is láttam olvashatóbb listákat!

Ennyit erről a könyvről. Ami most következik, az a többi BASIC-tárgyú izéről szól. Elöljáróban egy XX. századi népi bölcsességet szeretnék megosztani a nyilvánossággal: általában ne a szerzőt szidjuk, a hibákról ő tehet a legkevésbé. A lektor is felelős, de leginkább a kiadó.

Hámori Miklós nem túl olvasmányos, talán unalmas könyvét (1) próbálta feldobni egy pici BASIC-betéttel. 32 oldalon foglalja össze a tudnivalókat, látszólag géptől függetlenül, de érzésem szerint elsősorban ABC–80-ra gondolva (a könyv írásakor ez volt a legelterjedtebb típus). A „MM. rendeletére" készült mű sem sikeredett TAN-ra, bár ez kétszer is szerepel a címben.

A HT-hez három év alatt sem jött össze elfogadható segédanyag. A gyári kiadványról (2) már volt szó a BIT-LET 1983. novemberi VALLATÓ-jában – erre nincs mentség. A Tudományszervezési és Informatikai Intézet – menedzselte „HT kezdőknek" (3) semmivel sem jobb, ráadásul a „kezdőknek" címzés érthetetlen. A bevezetésben tudjuk meg, mi a különbség a BASIC-bővítés különböző belépési pontjai között. A „Parancsok" fejezet csak a CLOAD, RUN, LIST, CSAVE-ről tud, míg az „Utasítások" mindössze hatot tárgyal. Tömbök létéről a változók dimenzionálásakor értesülünk, jóval a FOR előtt. Ezúton tudatom a szerzőkkel, hogy a THEN/ELSE után utasítások is írhatók! READ-DATA miért előbbre való mint az INPUT?

Az Alcock–Ránky-féle kézírásos könyv (4) egyetlen komoly baja, hogy ez volt az első nagy példányszámú kiadvány. Második-harmadiknak zseniális, de kezdőknek nem merném ajánlani. Amúgy igen jó kézikönyv, kellemes stílussal. A sajtóhibák körüli huzavona, mondhatni szájtépés nevetséges: nem múlja felül lényegesen a hazai átlagot.

Kőhegyiék „nyelvjárásai" (5) még jobban tetszett, gyorsan követte példaadóját. Sajnos erősen támaszkodik is rá, így kezdőknek szintén nem optimális, jóllehet didaktikusabb. Nagy hiba, hogy a kiadó nem mert több könyv önálló megírására bátorítani. Szerzők meg tudtak volna birkózni a feladattal, sőt úgy tűnik, az eredetinél kedvezőbb szinten!

Bodor–Gerő lélegzetelállító alkotása (6) sok fejtörést okozott. Viszonylag korán készült, kifejezetten alacsony szintről indul és helyenként átgondoltság látszatát keltve halad. Nos, rejtély, hogy hova, „Adatnevek", „Program folyamat", „Kivezérelt feltételszerkezetek" fejezetcímek csak a számítástechnikai jogászokat nem riasztják el. Szintén nem laikusoknak való a FOR-t megelőző ON, GOSUB, AND-OR stb. illetve a töméntelen blokkdiagramszerű ábra. A „Programozási konvenciók" a végefelé viszont egy nyári tábor nebulóinak is unalmas. Esetszétválasztásra sokkal szebb példa egy barkochba, mint a visszaköszönő másodfokú egyenlet. Nem vagyok kezdő, de a „Soros karbantartás", és „Nagygépes szimuláció" részekkel nem tudtam megbirkózni. Maliciózusan összevetve az ugyancsak kétszerzős „Hetedhét"-tel, úgy tűnik, több segítség egy laikus, mint egy szakember?!

A Kocsis-trilógia harmadik kötete kézikönyv lévén, csak a másik kettőről (7) essék szó. A nyelvezet ismét a számítástechnikai jogászokat idézi. Rejtély, hogy Pólya György hazájában hogyan lehet a „feladatmegoldásról", a 3. fejezet stílusában írni! Tragikusan keverednek a BASIC- és középgépes operációs rendszer funkciók. Sőt félek, az ajánlás ellenére („Bálintnak és Gábornak"), a PC-k korában a könyv hallgatólagosan közép- és nagygépek használóinak íródott. Piranha rajként hemzsegnek a „problémák modulszerkezetei". Figyelemre méltó az első öt („kedvcsináló") példaprogram:

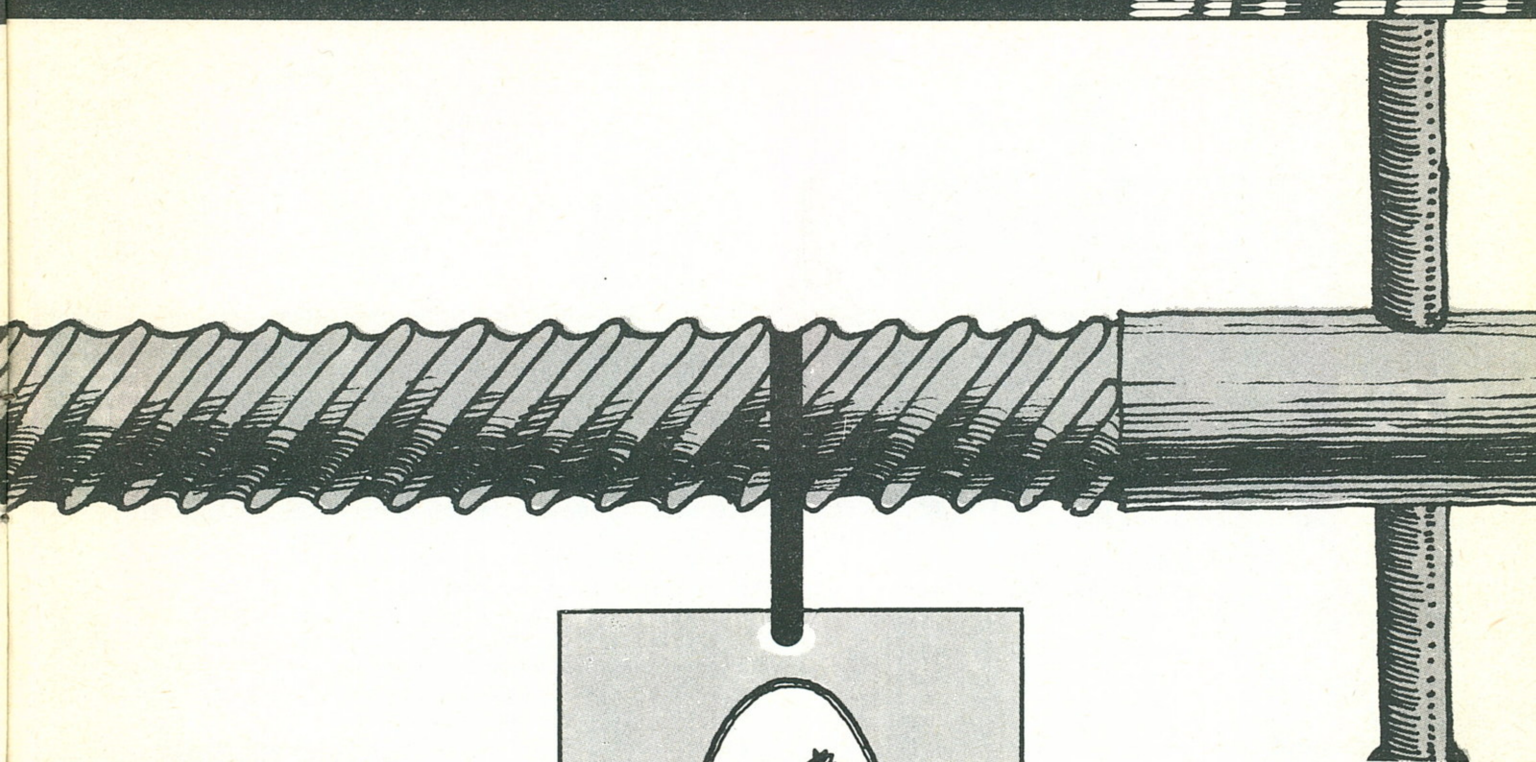
ellenállás-számítás – a 17 sorból 8 REM, 6 LET...
beruházás-gazdaságosság – kb. 35 sorból 12 REM, 14 PRINT
amortizációszámítás – AMOR neve miatt véletlenül megnéztem. l. később
bérelszámolás
címletösszetétel meghatározása kifizetőhelyen.

AMOR-ból egy pajzán részlet 115. oldal):

```
220 IF K<5 THEN GO TO 240
230 GO TO 270
240 REM * THEN AG *
250          N=(B*(5-K))/5
260          GO TO 290
270 REM * ELSE AG *
280          N=1.
290 REM * IF VEG *
```

Részlet kibogozása és rövidítése házi feladat kezdőknek. A BASIC amúgy sem túl strukturált nyelv – nehéz jól áttekinthető listájú programot írni –, de ez azért szándékos hitelrontás!

Úry László első kötete (8) a szivárványszínű sorozat sikeres darabja, egy elég jó kézikönyv. Nem tankönyv, így a módszernél fontosabb



a teljessége. Picit szerencsétlennek tűnik az utasítások névsorban történő ömlesztése, illetve nagyon hiányzik egy tárgymutató. Igen szűkszavú a SIMON'S BASIC-ről szóló rész, elég összevissza, és pongyola. (Lehet, hogy egy rossz fordítás?) Tetszett viszont az utasítások tárgyalása: tények-példák-magyarázat. Végeredményben lényegesen jobbnak tartom, mint a géphez adott, gyári „USER MANUAL"-t. Lényegében hasonló következtetésre jut Turchányi Géza az INFORMÁCIÓ elektronik 85/3. számában.

Az IIK dicséretes buzgalommal fordítgatja az idegen nyelvű gépkönyveket (9). Sem a szellemi befektetés, sem a primitív nyomdatechnika nem indokolja a szerintem horribilis árat. Lehet, hogy a temérdek sajtóhibát kell megfizetnünk?

A TV-BASIC című torzszülöttre (10) vagyok a legmérgesebb! Szerzőnek sikerült a távolról sem tökéletes háromkötetes művét (7) alulmúlni. Ráadásul ez a TANFOLYAM segédkönyve!? Rendkívül zavaró a négy gép HT, ZX, PRIMO, C64 – egyszerre történő tárgyalása. A nyelvezetéről lásd (6) és (7). Ez a stílus a Közös kiadó hivatalos nyelve? De miért kínoznak vele egy országot? Még mindig nem tudom pontosan, mi a „modul", de a helyőr szerzett néhány vidám percet. A 79. oldalon olvastam: „A tesztelés lényege, hogy különböző bemeneti adatokkal próbáljuk ki a programot (illetve a modulokat), és közben figyeljük, hogy a specifikáció szerint működik-e". Eddig sok olvasónak illene eljutni, lévén ez a 4. adás: A „több programból álló szerkezetek" súlyáról vajon mit lehet tudni?

Aki az első fejezetben a billentyűzettel ismerkedik, pár hét múlva adatállományokat szervez a 13–14. fejezet szintjén. Megértem, hogy 120 forintos könyvet úgy lehet a legtöbb emberre rásózni – a tv-adás propagandájával – ha meg se próbáltuk tisztázni, kinek szól. Az már más kérdés, kit érdekel, lesznek-e a vásárlókból olvasók? A nagy buzgalommal szervezett országos ÜGY ezen a ponton többet érdemelt volna.

Eredetileg csak egyetlen könyvről akartam írni, közben fogalmazódtak meg a többiről mondottak. Most megint úgy érzem, tovább lehet/kell menni. Többször hallhattunk, olvashattunk például a BIT-LET-ben is – a számítógépeket övező ködös misztikáról. Nem erősíti-e a jelen írásból kibontakozó kiadói, sőt általában tömegkommunikációs politika ezt a jelenséget? Jóllehet ez egyáltalán nem kívánatos, de sokaknak igen kellemes, sőt jövedelmező.



1. kín:
ár – 4,5
(Önmagában-többihez képest)
2. kín:
perifériák – 5
(Egyetlen rossz könyvre sem utal!)
3. kín:
képernyőkezelés – 5
(Sok nekifutásra, didaktikusan)
4. kín:
hang – 5
(Nagyon helyesen kimaradt!)
5. kín:
programtárolás – 3,5
(L. 3. fenntartás)
6. kín:
gépi kód – 5
(Egy szót se róla!)
7. kín:
megbízhatóság (sajtóhibák) – 4
(Néhány apróság pl. 123., 140., 327. old.)
8. kín:
billentyűzet – 4,5
(L. 2. fenntartás)
9. kín:
dokumentáció –
10. kín:
editálás – 4,5
(L. 1. fenntartás)
11. kín:
nyelve(zete) – 5
(L. TETSZETT)
12. kín:
tanulhatóság – 5
(Próbáld ki!)
13. kín:
emberközelség – 4
(L. 6. fenntartás)
+ ***1. kín:***
szubjektív – 5

átlag: 4,6

Minek minősítsük, hogy az „Iskolaszámítógép program" évi sok-sok (talán száz) milliójából nem tellett még egyetlen elfogadható alkotásra. A Tankönyvkiadó sem tekinti ezt a feladatának vagy hiányzik az „MM rendelete"? Úgy érzem pedig lenne rá kapacitás, de a kiadók a bólogató szerzőt szeretik, aki kurtításba, egyszerűsítésbe beleegyezik, tartja a határidőket stb. Több könyv(tervezet)ről tudok, ami valahol fekszik. Lőcs Gyula – aki már jó tíz éve letette névjegyét néhány nyelvkönyvvel – érdekes sorozata a „Számítástechnika" folyóiratban nem érdemelne meg egy átdolgozást könyvvé? Sajnos, már átírás elképzelhető csak el – gondolva a közben eltelt 4–5 évre. Dusza Árpád – a hazai számítógépes oktatás egyik „nesztora" – tankönyv-kéziratai miért fiókokban porosodnak? Vajon Kovács Mihály (Bp. Piarista Gimnázium) tapasztalatai és eredményei nem érnének meg egy írást? Az ELTE ABC-80 füzeteit nem lenne érdemes más típusra is átdolgoztatni és kiadni? A KFKI-ban is jelent már meg pár dolog, amit ismer (hetne) néhány kiadó. Ezek a művek mind ismerőseim, és valószínűleg sokszor ennyi az elfekvő készlet. Sajnos a minőség feltétele a válogatás.

Nem tíz, de talán száz könyvet el lehetne sütni ebben, és hasonló témakörökben – mint ahogy a piacfüggő nyugati könyvkiadás tanú rá. Nálunk a szerzőn és olvasón kívül senki nem érdekelt ebben. Sajnos ők nem tudnak könyvet csinálni. Még legalább egy láncszem hiányzik.

Török Turul

(1) Dr. Hámori Miklós: Tanulás és tanítás számítógéppel. – Tankönyvkiadó, 1983.
(2) HT–1080Z – BASIC-programozás. – Híradástechnika Szöv. 1983.
(3) Utassy–Mersich–Székely: HT 1080Z kezdőknek I. – TII 1984.
(4) D. Alcock: Ismerd meg a BASIC-nyelvet! – Műszaki K. 1983.
(5) Szerk. Kőhegyi János: Ismerd meg a BASIC nyelvjárásait! – Műszaki Kiadó 1984.
(6) Bodor–Gerő: A BASIC-programozás technikája. – SZÁMALK. 1984.
(7) Dr. Kocsis András: Programozás BASIC nyelven I–II. – SZÁMALK. 1984.
(8) Dr. Úry László: COMMODORE 64. I. – LSI. 1984.
(9) ZX 81 és ZX SPECTRUM – IIK. 1984.
(10) Dr. Kocsis András: TV-BASIC – SZÁMALK. 1984.



SZAKKÖRÖKNEK!

LEBEGŐPONTOS ARITMETIKA A SPECTRUMON V.

A mostani alkalomra már csak egy néhány kód jelentésének tisztázása maradt. Ezek közül az első négy olyan, hogy elsősorban saját célokra használja az interpreter.

39H A szögfüggvények számításánál használatos. Híváskor a stack tetején álló számot jelöljük X-szel, visszatéréskor ugyanott legyen V! X-ből képezzük a V számot, hogy

i. $-1 \leq V \leq 1$ legyen és
ii. SIN(PI*V/2)=SIN X teljesüljön.

(Azaz a szögfüggvények sorbafejtéséhez van szükség erre a műveletre.)

A transzformáció két lépésben történik:

- Képezzük az alábbi kifejezés értékét:
  Y=4*(X/(2*PI)−INT(X/(2*PI)+.5))
- V = Y, ha $-1 \leq Y \leq 1$
  V = 2−Y, ha $1 < Y < 2$
  V = −2−Y, ha $-2 \leq Y < -1$

Matematikai módszerekkel belátható, hogy az X és az így kapott V értékre teljesülnek a fent leírt feltételek.

3CH $X*10^M$ alakban adott szám értékét számolja ki. X híváskor a stack tetején áll, M (pozitív vagy negatív egész) az akkumulátorban van.

3DH „Kis egész" formában ábrázolt számot az általános 5 byte-os alakra konvertál. (Nem mondtuk ki ugyan határozottan, de a kis egészeket [l. januári BIT-LET] *nem kötelező* más módon ábrázolni, mint a többi számot.)
Ezt a rutint a logaritmus-, exponenciális stb. függvények hívása előtt használja az interpreter.

1AH Egy byte-ot beolvas az adott csatornáról. A csatorna számát a stack tetején álló szám adja meg. Erről bővebbet most nem írunk, ugyanis sok olyan ismeretet tételez fel a használata, amely most nagyon messzire vezetne.
Látható, hogy ezekkel a műveletekkel lényegében minden olyan tevékenység elvégezhető, ami BASIC-ben. Azonban sokakban felmerülhet, hogy miért van erre mind szükség, hiszen néhány, jól kiválasztott alap-rutin segítségével a többiek összeszervezhetők lennének. Természetesen a Spectrum interpreterének írói éppúgy tisztában voltak ezzel, és tényleg csak egynéhány rutin van megírva, a többi ezek összeszervezéséből áll. Ez utóbbiak arról ismerhetők fel, hogy egy RST 28H utasítással „belülről" újra meghívják a lebegőpontos kalkulátort. (Ez a rekurzió *az egyik* oka annak, hogy a sok számítást igénylő BASIC programok viszonylag lassan futnak a Spectrumon [l. 1984. januári BIT-LET, Benchmark.])
Ez azonban egy korábbi megjegyzésünk magyarázatául is szolgál: az újabb RST 28H természetesen elronthatja a BREG tartalmát, amely kizárja ezen utasításoknak ilyen belső ciklusban történő használatát. Most éppen ezért összefoglaljuk azon rutinok kódjait, amelyek *nem rontják el* a BREG-et:

00H–05H, 07H–0FH, 17H–1EH, 29H–31H, 33H–37H, 3AH, A0H–A4H, C0H–C5H, E0H–E5H.

Mielőtt befejeznénk az ismertetést, szólnunk kell még egy kódsorozatról: **82H–9FH.** Ezek segítségével egy egész polinom helyettesítési értékét számoltathatjuk ki. Használata lényegében egyszerű: a kód után leírjuk az együtthatókat (ugyanúgy, mintha a 34H kódot használnánk [l. a sorozat harmadik részét, júniusi BIT-LET], de a 34H-t természetesen nem írjuk le). A független változót a stack tetején adjuk át. Az együtthatók darabszámát a kódból úgy kaphatjuk meg, hogy a kódból kivonunk 80H-t. A bonyolultságot mégis az okozza, hogy a felsorolásban *nem* a tényleges polinom együtthatóit kell megadni. Ez a rutin ugyanis a független változó ún. *Csebisev-polinomjával* számol.

Ez az utóbb említett módszer ismertetése túlmenne a jelen sorozat keretein, de azok az olvasók, akik kedvet éreznek (és alapképzettségük is megvan) hozzá, már el tudnak indulni egy felfedező úton kikísérletezni, hogyan is megy ez pontosan.

Tudjuk, hogy ez a sorozat nem lehet teljes, hiszen az egész aritmetikáról nemhogy cikksorozatot, hanem egész könyvsorozatot írhatna az, aki minden apró lehetőséget, érdekességet fel akar tárni. Sorozatunkkal leginkább azt akartuk bemutatni, hogy a ROM-rutinok közül gyakorlatilag mindent fel lehet használni saját célokra is, és például nem kell feltétlenül egy gépi kódú programból csak azért visszatérni BASIC-be, mert bizonyos számításokat ott tűnik kényelmesnek elvégezni. Gépi kódban maradva természetesen lesz egy kis többletmunkánk, de ne felejtsük el, hogy ez is jár olyan és annyi előnnyel, mint bármely más gépi kódú program: sokkal gyorsabb, és kevesebb tárkapacitást igényel.

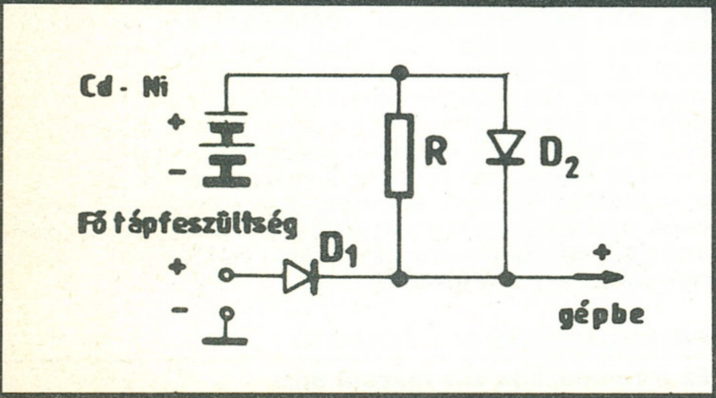
Reméljük, hogy sorozatunkat sokan jól fogják tudni hasznosítani későbbi programjaikban.

Halász Péter

POSTA

Tisztelt Szerkesztőség!
Sok bosszúságot okozott már programozáskor az áramkimaradás. Találtam egy kapcsolást, amely használatával nem vész el a program áramkimaradáskor, de sajnos konkrét adatokat nem tartalmaz. Kérem, adjanak tanácsot, hogyan lehetne megépíteni?
Szabó József
Balmazújváros, Debreceni út 3–5.

A válaszadáshoz régi ismerőseinket, az ELKON GM-et kértük, segítsenek. Az alábbi választ az ő levelük alapján írjuk.
A közölt kapcsolási rajz valóban véd áramkimaradás esetére, de a Spectrum viszonylag nagy áramfelvétele (700–800 mA) még rövid idejű tápellátási igény esetén is meglehetősen nagy méretű és kapacitású akkumulátort igényel. Éppen ezért a szünetmentes táplálás csak ott indokolt, ahol a rendszer folyamatos működése elengedhetetlen (pl. folyamatirányítás).
A külső tápforrás 9–10 V közötti legyen, azaz 8 db R14 (Baby) vagy R20 (Góliát) méretű Ni–Cd akkumulátor szükséges kb. 30 perces áramkimaradáshoz.
A két dióda legalább 1 A-es legyen, az R ellenállás a töltőáramot hivatott korlátozni.
Mindehhez szerkesztőségünk megjegyzése:
Szerintünk a probléma ilyen megoldása csak elméleti jelentőségű, mert a folyamatos programozáshoz úgyis szükség van még egy működő tv-re. Annak tápellátását ezzel a módszerrel megoldani nem lehet, lévén a tv-nek a Spectruménál is lényegesen nagyobb áramigénye (pl. a Junoszty tv 12 V-ról kb. 3–3,5 A). Éppen ezért, ha a tv-t autóakkuról üzemeltetjük, akkor arról megoldható a Spectrum üzemeltetése is, csak arra kell ügyelni, hogy az autóakku feszültsége a Spectrumnak egy kicsit sok.
A levélben szereplő egyéb kérdésekre levélben válaszoltunk.



Tisztelt BIT-LET!
3. osztályos szakközépiskolás vagyok. Eddig az iskolában foglalkoztam számítástechnikával, és elég jól sikerült elsajátítanom a BASIC-nyelvet. Nemrég hozzájutottam egy ORIC–1 típusú 16 K-s mikroszámítógéphez. Nagyon örültem neki, azonban ez a gép hazánkban nem túl ismert, és a hozzá mellékelt angol és német nyelvű gépkönyvet sem sikerült lefordítanom. Így csak a mintaprogramokat gépelgettem be, majd miután megismertem az ORIC egy-két sajátos utasítását saját programokat is kezdtem írni.
Szeretném teljesen megtanulni a gép BASIC-jét, és gépi kódú programozását, ehhez azonban magyar nyelvű leírás kellene. Ebben kérném az Önök segítségét: írják meg, hogy hol lehetne ilyet beszerezni, vagy ha lenne olyan olvasó, aki másolás céljából el tudna küldeni egy magyar nyelvű gépkönyvet, vagy valamilyen más úton át tudná adni tapasztalatait az ORIC-kal kapcsolatban. Programokat is szívesen cserélnék! Ezenkívül szeretném megtudni, hogy Magyarországon hol lehetne ORIC–1-hez vásárolni tartozékokat (pl.: memóriabővítőt, botkormányt stb.) és softvert
Ihász István
8200 Veszprém, Haszkovó u. 18/A

Mi nem küldünk Önnek semmit, de talán a márkatársak!

Tisztelt Szerkesztőség!
HT-használó vagyok, kérdéseim is ezzel kapcsolatosak.
1. Egy számítógépen megírt dalt lehet-e közvetlenül (csatlakozókábellel) felvenni műsoros kazettára vagy mindenképpen mikrofon szükségeltetik ehhez?
2. Hogyan lehet megakadályozni egy READY után következő parancs végrehajtódását (pl. LIST és LLIST kizárását)?
3. Az egyik előző BIT-LET-ben (85. máj. 30.) találkoztam egy számomra érthetetlen dologgal.
Idézem (32. o. 3. pont):
„... ha egy kód egymás után többször fordulna elő, akkor például eléírjuk mínusz előjellel azt, hogy hány darab kell belőle, s utána egyszer leírjuk a kódot.
Hogyan lehetséges ez? A cikkben erre nincs magyarázat, programfejtegetéseim pedig nem jártak sikerrel ez ügyben. De ha ez mégis így igaz, akkor miért van az első példaprogram 900-as sorában a 140-es és a 25-ös érték előtt —1? Ez megkérdőjelezi az egésznek az értelmét!
4. Programlistát (játék) nyomtatón elkészítve elfogadnak-e és fizetnek-e érte honoráriumot? Ha igen, értesítsenek ennek módjáról!
Fülöp Ferenc
Budapest, Prieszol J. u. 2. 1203

1. A HT-n van lehetőség közvetlenül, csatlakozókábellel felvenni magnóra a generált zenét. A gép hátán levő csatlakozók kiosztását az új kiadású gépkönyv közli.
2. Már megírtuk, hogy ilyen „drasztikus" programvédelemmel nem foglalkozunk, bár a módszert ismerjük. (Hogy ezt milyen könnyű kijátszani leírjuk, hogy pl. PRINT:LIST-et már végrehajt még emellett a védelem mellett is.)
3. Az említett cikkben nem derül ki egyértelműen, de a példák, feladatok előtti rész csak ötleteket ad, az első megvalósításnál ennek egy módosított változatát használta a szerző, amelyet le is írt. (A második program teljes egészében az Ön által is leírt elv alapján készült.)
4. Már többször megírtuk, hogy programokat k zettán kérünk, hogy kipróbálhassuk. Természetesen, ha jónak, ötletesnek találjuk (például „lövöldöző" játék nem valószínű, hogy ilyen), akkor közöljük, és a szokásos honoráriumot fizetjük érte.

Tisztelt Olvasóink!
Nem értjük miért, de egyre többen kérik, hogy kizárólag levélben válaszoljunk problémáikra. Úgy érezzük, hogy mi éppen azért vagyunk, hogy az olyan kérdésekre, amely valószínűleg sokakat érinthet, a lap hasábjain válaszoljunk. Éppen ezért kérjük, hogy minél kevesebben írják oda a levelük végére, hogy levélben kérnek választ!
A másik problémánkat már egyszer szóvátettük, de úgy látszik, nem lehet elégszer ismételni? Beküldött programokat csak abban az esetben tudunk közölni, ha azt kazettán is megkapjuk, ugyanis arra nincs kapacitásunk, hogy mi „bepötyögjük" valamely számítógépbe. Ugyancsak sokan küldenek be használati utasítás, programleírás nélkül programokat. Ezt is szóvá tettük néhányszor, hogy ezekkel nem tudunk mit kezdeni.
Kérjük Olvasóinkat, hogy ilyen hiányos formájú közlendőkkel minél kevesebbel, tartalmas, teljes anyagokkal minél többel örvendeztessenek meg bennünket!

Múlt havi BIT-LET-ünk Nyílt tér rovatában Lázár Károly levelét közöltük. Ebben többek között olvasónk szóvá tette, hogy nem jut megfelelő, a gépi kódú programozást tanító irodalomhoz. Ismeri ugyan lapunk sorozatát, de mint írta, kifejezetten Spectrum géphez szeretne anyagot. Lapunkat elolvasván kedves levelet írt hozzánk dr. Laczkó Béla a Mikroelektronika c. lap szerkesztője. Fölhívja olvasóink figyelmét a lapban megjelenő e témával foglalkozó sorozatra, amelyből eddig hat rész jelent meg, s további kettő előkészületben van. Köszönjük az információt, s azt is, hogy fölhívta figyelmünket lapjuk legújabb számának néhány írására, így a Tóth Ferenc által a HT 1080 Z programozásáról írott cikkére, valamint Pálfalvi Jenőnek a VHS rendszerű képmagnókról szóló írására.

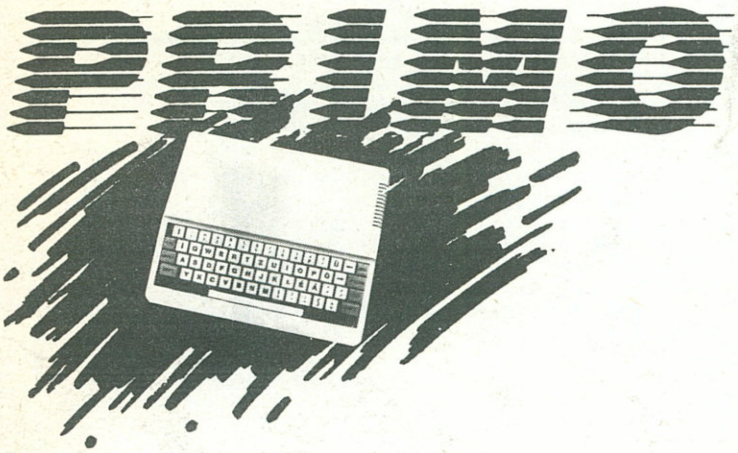
Őszinte megbánással tudatjuk, hogy múlt havi számunk 28–29. oldalán a Primó zene-bona című programajánlatunk szerző nélkül jelent meg. Tudatva, hogy nem a nyomda ördöge, hanem a feledékenység okából történt a dolog – ehelyütt pótlólag közöljük, hogy a szerző
Örley Gábor; Örley Gábor; Örley Gábor; Örley Gábor

A MICOLOR-NYERŐ PÁLYÁZAT VÉGEREDMÉNYE

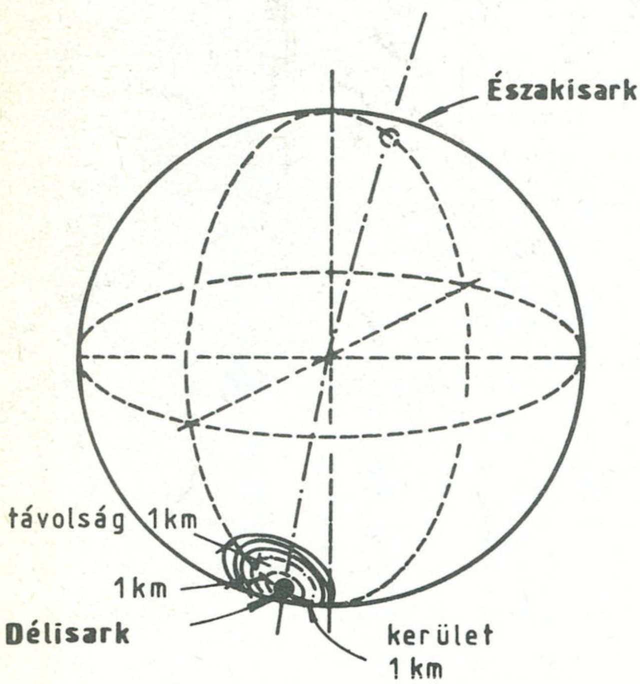
1. Piarista Gimnázium, Bp. 98 pont • 2. Berzsenyi Dániel Gimn. 3. szakkör Bp. 97 pont • 3. Földes Ferenc Gimn. A szakkör, Miskolc 92 pont • 4. Vörösmarty Mihály Gimn., Érd 87 pont • 5. Móricz Zsigmond Gimn., Tiszakécske 86 pont • 6. 1. sz. Ipari Szakmunkásképző Int. Miskolc 85 pont

Dicséret: JATE Ságvári Endre Gyak. Gimn. 2, 3 szakkör, Szeged • Katona József Gimn., Kecskemét • Leövey K. Gimn., Pécs • Móricz Zsigmond Gimn., Szentendre • Roth Gyula Erdőgazd. és Faip.-i Szakköz. isk., Sopron

Az első helyezett az ígért MICOLOR 1-et kapja, a 2. helyezett 1 éves előfizetést az Ötletre, a további közölt helyezettek oklevélben részesülnek.



• NYERŐ • NYERŐ •



A 2. FELADAT MEGOLDÁSA

A feladat közismert megoldása: az ember az Északi-sarkon állt. Azonban nem csak ez az egy megoldás van. Képzeljük el ugyanis az 1 km kerületű délkört (déli szélességi kör), s vegyük ennek a tetszőleges pontját, menjünk innen képzeletben északra 1 km-t. Induljon innen az emberünk! Ekkor dél felé menve 1 km-t, épp az említett délkörre jut, 1 km-t keletre menve visszajut ugyanide, s innen egy km-t északra menve épp a kiindulási helyére érkezik. Tehát az 1 km kerületű délkörtől 1 km-re északra fekvő délkör minden pontja jó kiindulási hely. De ugyanúgy jó a 1/2 km kerületű délkörtől 1 km-re északra fekvő délkör minden pontja is, az 1/3 km kerületűtől 1 km-re északra fekvő minden pontja is, és így tovább. Láthatjuk, hogy nemcsak végtelen sok kiindulási pont van, hanem végtelen sok délkör is van, melyeknek minden pontja jó kiindulási pont. Határhelyzetként jónak vehetjük a Déli-sarktól 1 km-re fekvő délkör pontjait is, bár ha emberünk innen indul, az első 1 km után a Déli-sarkra érkezik, s itt igazából nem tud keletre menni.

Más megoldás viszont nincs. Ugyanis az 1 km, 1/2 km stb. kerületű északi szélességi kör az Északi-sarkhoz 1 km-nél közelebb van, így ezekre nem tudunk úgy eljutni, hogy valahonnan 1 km-t délre megyünk. Ha pedig a keletre menés során igazából helyet változtatunk, akkor egy másik hosszúsági kör mentén kell északra mennünk, mint amelyik mentén délre mentünk, s a hosszúsági körök csak a sarkokon metszik egymást (itt hosszúsági körön csak a sarktól sarkig terjedő félköröket értjük, mint ahogyan így is számozzák őket). A Déli-sark pedig nem lehetett a kiindulási pont.

PRIMÓ-NYERŐ 3. FELADAT

Ez a feladatunk két részből áll:

a) Van-e olyan derékszögű háromszög, melynek oldalhosszúságai egész számok, s valamely két oldala négyzetének a különbsége 15?

b) A „Mohács" nevű magyar hajó a Földközi-tengeren hajózik. A hajó szélessége 35 méter. A hajó hosszának, a hajó súlyának és a kapitány életkorának a szorzata éppen az egész számok összege 1-től a hajóra festett évszámig (ezt eláruljuk: 1526). Kérdés: hány éves a kapitány?

Megjegyzés: ne felejtsék el kedves versenyzőink, hogy megoldásaikat meg is kell indokolni!

HÁNY ÉVES
A KAPITÁNY?

Kérjük levágni és a levélre felragasztani!
Beküldési határidő szeptember 15.!

AMINT ÍGÉRTÜK

augusztus 5-én délben ebéd helyett kisorsoltuk az előző Primo nyerő pályázat legjobbjai közt a gépet. Meglepően sokan – szám szerint nyolcan – jöttek el az érintettek közül, hogy jelen legyenek, s esetleg üdvrivalhassanak. Ez azonban ezúttal sem jött össze. Az eddigi sorsolásokon még nem fordult elő, hogy a nyertes ott lett volna! Sajnos!

A nyertes: **BERÉNYI ANDRÁS** Kaposvárról.

Kérjük jelentkezzen telefonon szerkesztőségünkben!

BIT-LET
Szerkesztőségünk címe: ÖTLET Budapest 1986. Szerkesztő: Angyalosi László. Művészeti szerkesztő: Pribelszky Pál
Tördelő: Domokos Imre. Hírrovat: Csányi György. Vallató: Pogány György. Sorvezető: Székely Jenő. Posta: Halász Péter